Local Termination: theory and practice

Endrullis, J.; de Vrijer, R.C.; Waldmann, J.

published in
Logical Methods in Computer Science
2010

DOI (link to publisher)
10.2168/LMCS-6(3:20)2010

document version
Publisher's PDF, also known as Version of record

Logical Methods in Computer Science
Vol. 6 (3:20) 2010, pp. 1–37
www.lmcs-online.org

Submitted Oct. 30, 2009
Published Sep. 7, 2010

LOCAL TERMINATION: THEORY AND PRACTICE

JÖRG ENDRULLIS [a], ROEL DE VRIJER [b], AND JOHANNES WALDMANN [c]

[a,b] Vrije Universiteit Amsterdam, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
e-mail address: joerg@few.vu.nl,rdv@cs.vu.nl

[c] Hochschule für Technik, Wirtschaft und Kultur Leipzig, Fakultät IMN, PF 30 11 66, D-04251 Leipzig, Germany
e-mail address: waldmann@imn.htwk-leipzig.de

Abstract. The characterisation of termination using well-founded monotone algebras has been a milestone on the way to automated termination techniques, of which we have seen an extensive development over the past years. Both the semantic characterisation and most known termination methods are concerned with *global* termination, uniformly of all the terms of a term rewriting system (TRS). In this paper we consider *local* termination, of specific sets of terms within a given TRS.

The principal goal of this paper is generalising the semantic characterisation of global termination to local termination. This is made possible by admitting the well-founded monotone algebras to be partial. We also extend our approach to local relative termination.

The interest in local termination naturally arises in program verification, where one is probably interested only in sensible inputs, or just wants to characterise the set of inputs for which a program terminates. Local termination will be also be of interest when dealing with a specific class of terms within a TRS that is known to be non-terminating, such as combinatory logic (CL) or a TRS encoding recursive program schemes or Turing machines.

We show how some of the well-known techniques for proving global termination, such as stepwise removal of rewrite rules and semantic labelling, can be adapted to the local case. We also describe transformations reducing local to global termination problems. The resulting techniques for proving local termination have in some cases already been automated.

One of our applications concerns the characterisation of the terminating S-terms in CL as regular language. Previously this language had already been found via a tedious analysis of the reduction behaviour of S-terms. These findings have now been vindicated by a fully automated and verified proof.

1. Introduction

An important contribution to the development of automated methods for proving termination has turned out to be the characterization of termination using well-founded monotone algebras. Both the semantic characterization and most known termination methods are concerned with *global* termination, uniformly of all the terms of a TRS. This is remarkable,

1998 ACM Subject Classification: D.3.1, F.4.1, F.4.2, I.1.1, I.1.3.

Key words and phrases: local termination, monotone algebras, combinatory logic, recursive program schemes.

DOI:10.2168/LMCS-6 (3:20) 2010

as termination is prima facie a property of individual terms. More generally, one may consider the termination problem for an arbitrary set of terms of a TRS. We call this the *local* termination problem.

A typical area where termination techniques are applied is that of program verification. The termination problems naturally arising in program verification are local termination problems: the central interest is termination of a program when started on a valid input. A simple example of a program that is not globally terminating is the factorial function:

$$\begin{aligned} \mathit{fac}(0) &= 1 \\ \mathit{fac}(n) &= n \cdot \mathit{fac}(n-1) \end{aligned}$$

This function terminates for all integers $n \geq 0$. However, when started on a negative number this function is caught in an infinite recursion. (This program will be used as an illustration in Examples 3.8 and 5.3.)

In logic programming (e.g. Prolog), local termination has been a central field of research over the past years. Local termination problems of Haskell programs have been considered in [PSS97] and [GSTSK06]. In [PSS97], a tableau calculus is devised to show termination of sets of terms of the form $f\, a_1 \ldots a_n$ where the a_i's are in normal form. In [GSTSK06], a transformation from Haskell programs into dependency pair problems [AG00] is given, which then in turn are solved using methods for global termination.

Surprisingly, for TRSs not much work is known about local termination. We mention the method of match-bounded string rewriting [GHW04], which can be used to prove local termination for sets of strings generated by a regular automaton. Indeed, this method can be viewed as an instance of the semantic framework we develop in this paper.

Local termination is of special interest when dealing with specific classes of terms within a TRS that is known to be non-terminating. Examples of such TRSs are combinatory logic (CL) [Cur30] and encodings of recursive program schemes or Turing machines. The well-known halting problem for Turing machines is a local termination problem. Clearly, this holds for the blank tape halting problem which just asks for termination on the blank tape. On the first glance the uniform halting problem – asking for termination on all inputs – might seem to be global. However, this is a local termination problem as well, since Turing machines are started in a distinguished initial state and admit only one head to work on the tape. In this paper we will use CL and the halting problem for Turing machines to illustrate some of our results (Examples 3.7, 6.5, 7.9, 7.10, 9.6 and 9.7).

Outline and Contribution. In Section 3 we generalize the semantic characterization from global termination to local termination based on well-founded, monotone partial Σ-algebras. This establishes a first, important step towards the development of automatable techniques for proving local termination. In Section 4 we extend this to relative termination, obtaining a characterization using extended monotone partial Σ-algebras.

For global termination it is common practice to stepwise simplify the proof obligation by removing rules. For local termination (the strictly decreasing) rules cannot simply be removed as they influence the set of reachable terms. We need to impose weak conditions on the 'removed' rules, see Section 5.

Having developed the general framework, in the remaining sections we look for fruitful instances of partial monotone algebras, suitable for automation.

In Section 6 we consider the case that the family of the set of terms for which we want to prove local termination can be described by a partial model. A variant of semantic

labeling [Zan95] can then be used to transform the local termination problem into a global termination problem, and the available provers for global termination can be applied.

In Section 7 we consider TRSs with the property that strong and weak normalization coincide. In particular, this holds for orthogonal, non-erasing TRSs. In case the language of normalizing terms happens to be regular, we show how a tree automaton (partial model) can be found accepting exactly the normalizing terms. Then we label the TRS with the obtained partial model, and employ the theory developed in Section 6 to transform the local termination problem for the set of normalizing terms to global termination of the labeled TRS. We automated the search for the tree automaton as well as the labeling.

We apply this method to two well-known combinators from CL: S and δ with the rewrite rules $\mathsf{S}\,xyz \to xz(yz)$ and $\delta\,xy \to y(xy)$, respectively. Determining the language N of normalizing S-terms has been open until the year 2000 [Wal00]. Using the method from Section 7 we can now automatically find the partial model for N, and we obtain a labeled system whose global termination coincides with local termination on N. Global termination of this labeled system (containing 1800 labeled rules) has been proven by TTT2 (1.0) [KSZM09] and the proof has been verified by CeTA (1.05) [TS09].

In Section 8 we demonstrate that the local termination method proposed in Section 6 can also be applied for proving global termination. To that end, we transform the global termination into local termination for the set of right-hand sides of forward closures [Der81]. Then we transform the obtained system back into a global termination problem using the transformation from Section 6. We show the applicability of this method by solving an example that remained unsolved in the last termination competition [Ter08]. After the transformation, the system allows for a simple termination proof using linear polynomial interpretations.

In Section 9 we combine the partial variant of the quasi-models of [Zan95] with monotone algebras to obtain partial monotone algebras. Roughly speaking, partial quasi-models are deterministic tree automata [CDG$^+$07] equipped with a relation $\geq$ on the states which guarantees that the language of the automaton is closed under rewriting. Thereby we obtain partial monotone algebras that can be applied successfully for proofs of local termination. Indeed, this method can be automated and, as a matter of fact, we have devised an implementation.

A preliminary version of this paper has appeared in [EdVW09]; our additional contribution is as follows:

- For TRSs where strong and weak normalization coincide and where the language of normalizing terms N is regular, we describe an algorithm for constructing a tree automaton (a partial model) accepting exactly the language N. For example, this method is applicable for (fully automatically) determining the language of normalizing S-terms.
- We show that methods for local termination can fruitfully be employed for proving global termination, classically the main focus of termination analysis for TRSs. Employing the RFC method (right-hand sides of forward closures) in combination with the transformation from local to global termination from Section 6, we give a new proof for a string rewrite system (SRS) for which no proof had been found in the termination competition so far.

2. Preliminaries

Term rewriting. A *signature* Σ is a non-empty set of symbols, each having a fixed *arity*, given by a map $\sharp : \Sigma \to \mathbb{N}$. Let Σ be a signature and $\mathcal{X}$ a set of variable symbols. The *set* $\mathsf{Ter}(\Sigma, \mathcal{X})$ *of terms over* Σ *and* $\mathcal{X}$ is the smallest set satisfying: $\mathcal{X} \subseteq \mathsf{Ter}(\Sigma, \mathcal{X})$, and $f(t_1, \dots, t_n) \in \mathsf{Ter}(\Sigma, \mathcal{X})$ if $f \in \Sigma$ with arity n and $\forall i (1 \leq i \leq n) : t_i \in \mathsf{Ter}(\Sigma, \mathcal{X})$. We use $x, y, z, \dots$ to range over variables. The set of positions $\mathcal{P}os(t) \subseteq \mathbb{N}^*$ of a term $t \in \mathsf{Ter}(\Sigma, \mathcal{X})$ is defined as follows: $\mathcal{P}os(f(t_1, \dots, t_n)) = \{\bot\} \cup \{ip \mid 1 \leq i \leq \sharp(f),\, p \in \mathcal{P}os(t_i)\}$ and $\mathcal{P}os(x) = \{\bot\}$ for variables $x \in \mathcal{X}$.

A substitution σ is a map $\sigma : \mathcal{X} \to \mathsf{Ter}(\Sigma, \mathcal{X})$. For a term $t \in \mathsf{Ter}(\Sigma, \mathcal{X})$ we define $t\sigma$ as the result of replacing each $x \in \mathcal{X}$ in t by $\sigma(x)$. Formally, $t\sigma$ is inductively defined by $x\sigma = \sigma(x)$ for variables $x \in \mathcal{X}$ and otherwise $f(t_1, \dots, t_n)\sigma = f(t_1\sigma, \dots, t_n\sigma)$. Let $[]$ be a fresh symbol, $[] \notin \Sigma \cup \mathcal{X}$. A *context* C is a term from $\mathsf{Ter}(\Sigma, \mathcal{X} \cup \{[]\})$ containing precisely one occurrence of $[]$. By $C[s]$ we denote the term $C\sigma$ where $\sigma([]) = s$ and $\sigma(x) = x$ for all $x \in \mathcal{X}$.

A *term rewriting system (TRS)* R over Σ and $\mathcal{X}$ is a set of pairs $\langle \ell, r \rangle \in \mathsf{Ter}(\Sigma, \mathcal{X})$, called *rewrite rules* and written as $\ell \to r$, for which the *left-hand side* ℓ is not a variable ($\ell \notin \mathcal{X}$), and all variables in the *right-hand side* r occur in ℓ as well ($\mathit{Var}(r) \subseteq \mathit{Var}(\ell)$). Let R be a TRS. For terms $s, t \in \mathsf{Ter}(\Sigma, \mathcal{X})$ we write $s \to_R t$ (or briefly $s \to t$) if there exists a rule $\ell \to r \in R$, a substitution σ and a context $C \in \mathsf{Ter}(\Sigma, \mathcal{X} \cup \{[]\})$ such that $s = C[\ell\sigma]$ and $t = C[r\sigma]$. The reflexive–transitive closure of $\to$ is denoted by $\twoheadrightarrow$. We call $\to$ the *one-step rewrite relation* induced by R and $\twoheadrightarrow$ the *many-step rewrite* or *reduction relation*. If $t \twoheadrightarrow t'$ then we call t' an *(R)-reduct* of t.

Definition 2.1. Let R be a TRS over Σ and $T \subseteq \mathsf{Ter}(\Sigma, \mathcal{X})$ a set of terms. The *family* $\mathcal{F}am_R(T)$ *of* T is the set of (not necessarily proper) subterms of R-reducts of terms $t \in T$ (that is, the least set containing t that is closed under reduction and taking subterms).

Partial functions. For partial functions $f : A_1 \times \ldots \times A_n \rightharpoonup A$ and $a_1 \in A_1, \dots, a_n \in A_n$ we call $f(a_1, \dots, a_n)$ *defined* and write $f(a_1, \dots, a_n)\!\downarrow$ whenever $\langle a_1, \dots, a_n \rangle$ is in the domain of f. Otherwise $f(a_1, \dots, a_n)$ is called *undefined* and we write $f(a_1, \dots, a_n)\!\uparrow$. We use the same terminology and notation for composite expressions involving partial functions. Between such expression we use Kleene equality:

$$\mathit{exp}_1 \simeq \mathit{exp}_2 \iff_{\mathit{def.}} (\mathit{exp}_1\!\uparrow \text{ and } \mathit{exp}_2\!\uparrow) \text{ or } (\mathit{exp}_1\!\downarrow \text{ and } \mathit{exp}_2\!\downarrow \text{ and } \mathit{exp}_1 = \mathit{exp}_2)$$

Note that an expression can only be defined if all its subexpressions are.

Definition 2.2. Let A be a set and R a relation on A. We define two properties of an n-ary partial function f with respect to R.

(1) f is *closed* if for every $a, b \in A$ we have:

$$f(\dots, a, \dots)\!\downarrow \;\&\; a\, R\, b \;\Rightarrow\; f(\dots, b, \dots)\!\downarrow$$

(2) f is *monotone* if for every $a, b \in A$ we have:

$$f(\dots, a, \dots)\!\downarrow \;\&\; f(\dots, b, \dots)\!\downarrow \;\&\; a\, R\, b \;\Rightarrow\; f(\dots, a, \dots)\; R\; f(\dots, b, \dots)$$

The functions that we consider will be typically both closed and monotone, which can be rendered briefly as:

$$f(\ldots,a,\ldots)\!\downarrow \;\;\&\;\; a\,R\,b \;\Rightarrow\; f(\ldots,a,\ldots)\,R\,f(\ldots,b,\ldots)$$

By writing something like $exp_1\ R\ exp_2$ we imply that exp_1 and exp_2 are defined.

Partial Σ-algebras. We give the definition of a partial algebra:

Definition 2.3. A *partial* Σ*-algebra* $\langle A, [\![\cdot]\!]\rangle$ consists of a non-empty set A and for each n-ary $f \in \Sigma$ a partial function $[\![f]\!] : A^n \rightharpoonup A$, the *interpretation of* f. A Σ*-algebra* $\langle A, [\![\cdot]\!]\rangle$ is a partial Σ-algebra $\langle A, [\![\cdot]\!]\rangle$ where all interpretations $[\![f]\!]$ for $f \in \Sigma$ are total.

Given a partial Σ-algebra $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ and a (partial) assignment of the variables, $\alpha : \mathcal{X} \rightharpoonup A$, we can give an *interpretation* $[\![t, \alpha]\!]$ of terms $t \in \mathsf{Ter}(\Sigma, \mathcal{X})$, which, however, will not always be defined. So the interpretation is a partial function from terms and partial assignments to A, inductively defined by:

$$[\![x, \alpha]\!] = \alpha(x)$$
$$[\![f(t_1, \ldots, t_n), \alpha]\!] = [\![f]\!]([\![t_1, \alpha]\!], \ldots, [\![t_n, \alpha]\!])$$

For ground terms $t \in \mathsf{Ter}(\Sigma, \varnothing)$ we write $[\![t]\!]$ for short.

Whenever a term t is defined, that is, $[\![t]\!]\!\downarrow$, then all subterms of t are defined as well. This is a consequence of the usual definition of the composition of partial functions (functional relations); 'undefined' is not an element of the domain. We say that a set of terms T is defined if all terms in T are defined:

Definition 2.4. A set $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$ is called *defined* if for all $t \in T$ we have $[\![t]\!]\!\downarrow$.

Partial models. First, we generalise the models from [Zan95] to partial models.

Definition 2.5. A *model* for a TRS R is a Σ-algebra $\langle A, [\![\cdot]\!]\rangle$ such that $[\![\ell, \alpha]\!] = [\![r, \alpha]\!]$ for every rule $\ell \to r \in R$ and every interpretation $\alpha : \mathit{Var}(\ell) \to A$ of the variables.

Definition 2.6. Let R be a TRS over Σ. A *partial model* $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ *for* R is a partial Σ-algebra $\mathcal{A}$, such that

$$[\![\ell, \alpha]\!]\!\downarrow \;\Rightarrow\; [\![\ell, \alpha]\!] = [\![r, \alpha]\!]$$

for every $\ell \to r \in R$ and $\alpha : \mathit{Var}(\ell) \to A$.

Thus the condition $[\![\ell, \alpha]\!] = [\![r, \alpha]\!]$ of models is only required if the interpretation of the left-hand side is defined, that is, $[\![\ell, \alpha]\!]\!\downarrow$. Observe that a left-hand side ℓ may be undefined while the corresponding right-hand side r is defined; the other way around is not permitted. This asymmetry is crucial, since rewriting may turn an undefined (non-terminating) term into a defined (terminating) term, but not the other way around.

Definition 2.7. Let $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ be a partial model. The *language* $\mathcal{L}(\mathcal{A})$ *of* $\mathcal{A}$ is:

$$\mathcal{L}(\mathcal{A}) = \{t \in \mathsf{Ter}(\Sigma, \varnothing) \mid [\![t]\!]\!\downarrow\}$$

We further generalise the concept of partial models to relations:

Definition 2.8. Let R be a TRS over Σ, $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ be a partial Σ-algebra, and $\succ \subseteq A \times A$ a binary relation. We say that $\langle A, [\![\cdot]\!], \succ\rangle$ is a *partial model for* R if:

$$[\![\ell, \alpha]\!]\downarrow \;\Rightarrow\; [\![\ell, \alpha]\!] \succ [\![r, \alpha]\!]$$

for all $\ell \to r \in R$ and every assignment $\alpha : \mathit{Var}(\ell) \to A$. If additionally $\langle A, [\![\cdot]\!]\rangle$ is a (total) Σ-algebra, then $\langle A, [\![\cdot]\!], \succ\rangle$ is called a *model for* R. Whenever the partial Σ-algebra $\mathcal{A}$ is clear from the context, we say '$\succ$ *is a partial model for* R' for short.

Note that $\langle A, [\![\cdot]\!]\rangle$ is a partial model if and only if $\langle A, [\![\cdot]\!], =\rangle$ is a partial model.

Remark 2.9. The notion of partial model $\langle A, [\![\cdot]\!], \succ\rangle$ is closely related to that of quasi-model [Zan95]. In particular, a quasi-model for a TRS R is a (total) monotone model $\langle A, [\![\cdot]\!], \geq\rangle$ for R.

Remark 2.10. In Definition 2.6 and 2.8 we could as well quantify over partial assignments $\alpha : \mathcal{X} \rightharpoonup A$ in place of total assignment $\alpha : \mathcal{X} \to A$. This gives rise to an equivalent definition as an undefined value for a variable in the left-hand side ℓ (and $\mathit{Var}(r) \subseteq \mathit{Var}(\ell)$) would result in $[\![\ell, \alpha]\!]$ being undefined, and thereby invalidate the precondition $[\![\ell, \alpha]\!]\downarrow$ of the implication.

3. Local Termination

We devise a complete characterization of local termination based on an extension of the monotone algebra approach of [EWZ08, Zan94]. The central idea is the use of monotone partial algebras, that is, the operations of the algebras are allowed to be partial functions. This idea was introduced in [EGH$^+$09], where these algebras have been employed to obtain a complete characterization of local infinitary strong normalization. First we give the definition of local termination:

Definition 3.1. Let A be a set and $\to \subseteq A \times A$ a binary relation on A. Then $\to$ is called *terminating on* $B \subseteq A$ if no $b \in B$ admits an infinite sequence

$$b = b_1 \to b_2 \to \dots$$

Note that $b = b_1 \in B$. The elements b_2, b_3, …, however, may or may not be in B.

Definition 3.2. A TRS R over Σ is called *terminating (or strongly normalizing) on* $T \subseteq \mathsf{Ter}(\Sigma, \mathcal{X})$, denoted $\mathsf{SN}_R(T)$, if $\to_R$ is terminating on T. We write SN_R for termination on the set of all terms $\mathsf{Ter}(\Sigma, \mathcal{X})$.

We introduce the concept of monotone partial Σ-algebras. In contrast with [EdVW09] we do not require well-foundedness of $\succ$. We think that it is conceptually cleaner to distinguish the two concepts. Monotone partial Σ-algebras $\langle A, [\![\cdot]\!], \succ\rangle$ for which well-foundedness of $\succ$ holds, will be called well-founded.

Definition 3.3. A *monotone partial* Σ*-algebra* $\langle A, [\![\cdot]\!], \succ\rangle$ is a partial Σ-algebra $\langle A, [\![\cdot]\!]\rangle$ equipped with a binary relation $\succ \subseteq A \times A$ on A such that for every $f \in \Sigma$ the function $[\![f]\!]$ is closed and monotone with respect to $\succ$.

A monotone partial Σ-algebra $\langle A, [\![\cdot]\!], \succ\rangle$ is called *well-founded* if $\succ$ is well-founded.

Remark 3.4. One could also work with monotone, *total* algebras instead of partial algebras, by adding an "undefined" element $\bot$ to the domain. Then defining $\bot$ to be maximal, $\bot \succ a$ for every $a \in A \setminus \{\bot\}$, monotonicity of a function will automatically entail closedness. In order to get full correspondence with our framework of partial algebras, we would in this set-up only consider strict functions (that is, the value of the function is $\bot$ whenever one of the arguments is $\bot$).

The following theorem gives a complete characterization of local termination in terms of monotone partial algebras.

Theorem 3.5. *Let R be a TRS over Σ, and $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$. Then $\mathsf{SN}_R(T)$ holds if and only if there exists a well-founded monotone partial Σ-algebra $\mathcal{A} = \langle A, [\![\cdot]\!], \succ \rangle$ such that T is defined, and $\succ$ is a partial model for R.*

Proof. Theorem 3.5 is proved in the same way as Theorem 4.6 (using Remark 4.2). □

To keep the presentation simple, the theorem characterizes local termination for sets of ground terms $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$ only. Indeed, the theorem can easily be generalized to sets of open terms by, instead of just a well-founded monotone partial algebra, additionally requiring a variable assignment α. A set of terms T is then called defined if for that α we have $[\![t, \alpha]\!]\downarrow$ for every $t \in T$.

Remark 3.6. In case $T = \mathsf{Ter}(\Sigma, \varnothing)$ is the set of all ground terms, Theorem 3.5 basically coincides with the usual theorem for proving termination using (total) well-founded monotone Σ-algebras. More precisely, the subalgebra of $\mathcal{A}$ containing all elements that are interpretations of ground terms (leaving out the junk) is a (total) well-founded monotone Σ-algebra proving termination of R.

Example 3.7. We consider the S combinator with the rewrite rule

$$\mathsf{S}xyz \to xz(yz)$$

from combinatory logic. That is:

$$@(@(@(\mathsf{S}, x), y), z) \to @(@(x, z), @(y, z))$$

in first order notation. The $@(M, N)$ is abbreviated by MN.

The S combinator is known to be globally non-terminating. For example the term $\mathsf{S}(\mathsf{SS})(\mathsf{SS})(\mathsf{S}(\mathsf{SS})(\mathsf{SS}))$ admits an infinite reduction, see further [Zac78]. We have, however, local termination on certain sets of terms, for example the set of "flat" S-terms:

$$T = \{\mathsf{S}^n \mid n \in \mathbb{N},\, n \geq 1\}$$

where $\mathsf{S}^1 = \mathsf{S}$ and $\mathsf{S}^{n+1} = @(\mathsf{S}^n, \mathsf{S})$.

We prove strong normalization on T using the well-founded monotone partial Σ-algebra $\mathcal{A} = \langle A, [\![\cdot]\!], \succ \rangle$, where $A = \{\mathsf{s}\} \cup \mathbb{N}$ and the interpretation $[\![\cdot]\!]$ is given by:

$$[\![\mathsf{S}]\!] = \mathsf{s} \qquad [\![@]\!](\mathsf{s}, \mathsf{s}) = 0 \qquad [\![@]\!](0, n) = n + 1 \qquad [\![@]\!](n, \mathsf{s}) = 2 \cdot n + 1$$

for all $n \in \mathbb{N}$ and $[\![@]\!](x, y)\uparrow$ for all other cases. Let $\succ$ be the natural order on $\mathbb{N}$; that is, s is neither source nor target of a $\succ$ step. Then well-foundedness of $\succ$ and monotonicity of $[\![@]\!]$ are obvious, and T is defined. We have $[\![\mathsf{S}xyz, \alpha]\!]\downarrow$ only if $\alpha(x) = \mathsf{s}$ and $\alpha(z) = \mathsf{s}$; then we obtain:

$$[\![\mathsf{S}xyz, \alpha]\!] = 3 \succ 1 = [\![xz(yz), \alpha]\!] \qquad \text{for } \alpha(y) = \mathsf{s}$$
$$[\![\mathsf{S}xyz, \alpha]\!] = 2 \cdot \alpha(y) + 3 \succ 2 \cdot \alpha(y) + 2 = [\![xz(yz), \alpha]\!] \qquad \text{for } \alpha(y) \in \mathbb{N}$$

Hence $\succ$ is a partial model for $\mathsf{S}xyz \to xz(yz)$ and we conclude termination on T.

Example 3.8. We recall the Haskell program from the introduction:

$$\begin{aligned} &fac(0) :: \mathsf{Integer} \to \mathsf{Integer} \\ &fac(0) = 1 \\ &fac(n) = n \cdot fac(n-1) \end{aligned}$$

We remark that the standard Haskell data type Integer allows for negative numbers. For this reason the program is not globally terminating, but only locally on non-negative integers. The usual implementation of the factorial function as TRS makes use of Peano numerals for encoding natural numbers using a constant '0' and a unary symbol 's' for successor. Then the problem of negative numbers does not occur.

For the purpose of modeling the Haskell program as close as possible, we have chosen for a different encoding of the factorial function as TRS. For encoding negative numbers we extend Peano numerals with a unary symbol '$-$'. Since standard term rewriting does not allow for a priority order on rules, we need to dissolve ambiguities, that is, overlaps between the rules, by instantiating the variables; e.g. for the factorial function fac the variable n needs to be instantiated with $\mathsf{s}(n)$ and $-(n)$ to match exactly the integers (in this case 0) not covered by the first rule. As the result of the translation we obtain the TRS R:

$$\begin{aligned} fac(0) &\to \mathsf{s}(0) && (\rho_1)\\ fac(\mathsf{s}(x)) &\to mul(\mathsf{s}(x), fac(x)) && (\rho_2)\\ fac(-(x)) &\to mul(-(x), fac(-(\mathsf{s}(x)))) && (\rho_3)\\ mul(x, 0) &\to 0 && (\rho_4)\\ mul(0, y) &\to 0 && (\rho_5)\\ mul(x, \mathsf{s}(y)) &\to add(mul(x, y), x) && (\rho_6)\\ mul(\mathsf{s}(x), -(y)) &\to -(mul(\mathsf{s}(x), y)) && (\rho_7)\\ mul(-(x), -(y)) &\to mul(x, y) && (\rho_9)\\ add(x, 0) &\to x && (\rho_{10})\\ add(0, y) &\to y && (\rho_{11})\\ add(x, \mathsf{s}(y)) &\to \mathsf{s}(add(x, y)) && (\rho_{12})\\ add(\mathsf{s}(x), -(\mathsf{s}(y))) &\to add(x, -(y)) && (\rho_{13})\\ add(\mathsf{s}(x), -(0)) &\to \mathsf{s}(x) && (\rho_{14})\\ add(-(x), -(y)) &\to -(add(x, y)) && (\rho_{15}) \end{aligned}$$

This TRS is globally non-terminating due to the rewrite sequence:

$$\begin{aligned} fac(-(x)) &\to mul(-(x), fac(-(\mathsf{s}(x))))\\ &\to mul(-(x), mul(-(x), fac(-(\mathsf{s}(\mathsf{s}(x)))))) \to \ \ldots \end{aligned}$$

We prove local termination on the set $T = \{fac(\mathsf{s}^n(0)) \mid n \in \mathbb{N}\}$. Let $\mathcal{A} = \langle \mathbb{N}, [\![\cdot]\!], > \rangle$ where $>$ is the natural order on $\mathbb{N}$, and the interpretation $[\![\cdot]\!]$ is given by:

$$[\![0]\!] = 0 \qquad [\![\mathsf{s}]\!](n) = n + 1 \qquad [\![fac]\!](n) = (2n+2)!$$
$$[\![mul]\!](n, m) = 2(n+1)(m+1) \qquad [\![add]\!](n, m) = n + 2m + 1 \qquad [\![-]\!](n, m)\uparrow$$

for all $n, m \in \mathbb{N}$. For all left-hand sides ℓ of (ρ_3), (ρ_7), (ρ_9), (ρ_{13}), (ρ_{14}), (ρ_{15}) and all $\alpha : \mathcal{X} \to \mathbb{N}$ we have $[\![\ell, \alpha]\!]\uparrow$; thus $>$ is a partial model for these rules. For the remaining rules we have:

$$[\![fac(0), \alpha]\!] = 2 > 1 = [\![\mathsf{s}(0), \alpha]\!]$$
$$[\![fac(\mathsf{s}(x)), \alpha]\!] = (2\alpha(x)+4)! = (2\alpha(x)+4)\cdot(2\alpha(x)+3)! >$$
$$2((2\alpha(x)+2)!+1)(\alpha(x)+2) = [\![mul(fac(x), \mathsf{s}(x)), \alpha]\!]$$
$$[\![mul(x, 0), \alpha]\!] = 2(\alpha(x)+1) > 0 = [\![0, \alpha]\!]$$
$$[\![mul(0, y), \alpha]\!] = 2(\alpha(y)+1) > 0 = [\![0, \alpha]\!]$$
$$[\![mul(x, \mathsf{s}(y)), \alpha]\!] = 2(\alpha(x)+1)(\alpha(y)+2) >$$
$$2(\alpha(x)+1)(\alpha(y)+1)+\alpha(x)+1 = [\![add(mul(x, y), x), \alpha]\!]$$
$$[\![add(x, 0), \alpha]\!] = \alpha(x)+1 > \alpha(x) = [\![x, \alpha]\!]$$
$$[\![add(0, y), \alpha]\!] = 2\alpha(y)+1 > \alpha(y) = [\![y, \alpha]\!]$$
$$[\![add(x, \mathsf{s}(y)), \alpha]\!] = \alpha(x)+2(\alpha(y)+1)+1 > \alpha(x)+2\alpha(y)+2 = [\![\mathsf{s}(add(x, y)), \alpha]\!]$$

for all $\alpha : \mathcal{X} \to \mathbb{N}$. Hence $>$ is a partial model for all rules in R. Moreover, T is defined (that is, $T{\downarrow}$) since $[\![fac(\mathsf{s}^n(0))]\!] = (2n+2)! \in \mathbb{N}$. By Theorem 3.5 we conclude $\mathsf{SN}_R(T)$, that is, R is terminating on T.

4. Local Relative Termination

We define local relative termination.

Definition 4.1. Let A be a set and $\to_1, \to_2 \subseteq A \times A$ binary relations. Then $\to_1$ is called *terminating relative to* $\to_2$ *on* $B \subseteq A$, denoted $\mathsf{SN}_{\to_1/\to_2}(B)$, if $\to_1 / \to_2 = \twoheadrightarrow_2 \cdot \to_1 \cdot \twoheadrightarrow_2$ is terminating on B. We write $\mathsf{SN}_{\to_1/\to_2}$ for relative termination on A.

Let R, S be TRSs over Σ, and $T \subseteq \mathsf{Ter}(\Sigma, \mathcal{X})$. Then the TRS R is called *terminating (or strongly normalizing) relative to* S *on* T, denoted $\mathsf{SN}_{R/S}(T)$, if $\to_R$ is terminating relative to $\to_S$ on T. We write $\mathsf{SN}_{R/S}$ for relative termination on all terms $\mathsf{Ter}(\Sigma, \mathcal{X})$.

Remark 4.2. Termination of R relative to S on T is equivalent to: no term $t \in T$ that admits an infinite rewrite sequence $t = t_1 \to_{R\cup S} t_2 \to_{R\cup S} \dots$ containing an infinite number of $\to_R$ steps. Furthermore we have $\mathsf{SN}_R(T)$ if and only if $\mathsf{SN}_{R/\varnothing}(T)$.

Definition 4.3. An *extended well-founded monotone partial* Σ*-algebra* $\langle A, [\![\cdot]\!], \succ, \sqsupseteq\rangle$ consists of monotone partial Σ-algebras $\langle A, [\![\cdot]\!], \succ\rangle$ and $\langle A, [\![\cdot]\!], \sqsupseteq\rangle$ such that $\mathsf{SN}_{\succ/\sqsupseteq}$ holds.

Note that $\mathsf{SN}_{\succ/\sqsupseteq}$ implies that $\langle A, [\![\cdot]\!], \succ\rangle$ is well-founded. The usual condition '$\succ \cdot \sqsupseteq \subseteq \succ$ and well-foundedness of $\succ$' is a special case of our condition '$\mathsf{SN}_{\succ/\sqsupseteq}$':

Lemma 4.4. *Let A be a set and $\sqsupseteq, \succ \subseteq A \times A$ binary relations such that $\succ$ is well-founded. Then $\succ \cdot \sqsupseteq \subseteq \succ$ implies $\mathsf{SN}_{\succ/\sqsupseteq}$.*

Proof. Assume that $\mathsf{SN}_{\succ/\sqsupseteq}$ would not hold. Then there exists an infinite $(\succ \cup \sqsupseteq)$-sequence containing infinitely many $\succ$ steps. Using $\succ \cdot \sqsupseteq \subseteq \succ$ we can remove all intermediate $\sqsupseteq$-steps giving rise to an infinite $\succ$-sequence, contradicting well-foundedness of $\succ$. □

Lemma 4.5. *Let $\langle A, [\![\cdot]\!], \succ, \sqsupseteq\rangle$ be an extended well-founded monotone partial Σ-algebra and let R and S be TRSs over Σ such that $\succ$ is a partial model for R, and $\sqsupseteq$ is a partial model for S. Furthermore, assume for $s \in \mathsf{Ter}(\Sigma, \varnothing)$ that $[\![s]\!]\downarrow$. Then we have the implications:*

(i) $s \to_R t \;\Rightarrow\; [\![s]\!] \succ [\![t]\!]$, *and*
(ii) $s \to_S t \;\Rightarrow\; [\![s]\!] \sqsupseteq [\![t]\!]$.

Proof. The proofs of (i) and (ii) are identical, we just prove (ii). Let $s \to_S t$, that is, we have a rule $\ell \to r \in S$, substitution σ and context C such that $s = C[\ell\sigma]$ and $t = C[r\sigma]$. Since $[\![s]\!]\downarrow$ and $\ell\sigma$ is a subterm of s, we also have $[\![\ell\sigma]\!]\downarrow$, so $[\![\ell, \alpha]\!] \sqsupseteq [\![r, \alpha]\!]$, as $\sqsupseteq$ is a partial model for S. Then using closedness and monotonicity of the interpretations $[\![f]\!]$ of all function symbols $f \in \Sigma$ we obtain $[\![s]\!] \sqsupseteq [\![t]\!]$. □

We give a complete characterization of local relative termination in terms of extended monotone partial algebras.

Theorem 4.6. *Let R and S be TRSs over Σ, and $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$. Then $\mathsf{SN}_{R/S}(T)$ holds if and only if there is an extended well-founded monotone partial Σ-algebra $\mathcal{A} = \langle A, [\![\cdot]\!], \succ, \sqsupseteq\rangle$ such that the set T is defined, $\succ$ is a partial model for R, and $\sqsupseteq$ is a partial model for S.*

Proof. For the 'only if'-part assume that $\mathsf{SN}_{R/S}(T)$ holds. Let $\mathcal{A} = \langle A, [\![\cdot]\!], \succ, \sqsupseteq\rangle$ where $A = \mathcal{F}am_{R\cup S}(T)$ and the interpretation of a function symbol $f \in \Sigma$ is defined by $[\![f]\!](t_1, \dots, t_n) = f(t_1, \dots, t_n)$ if $f(t_1, \dots, t_n) \in A$, and $[\![f]\!](t_1, \dots, t_n)\uparrow$ otherwise. The relations $\sqsupseteq$ and $\succ$ are defined by $\sqsupseteq = \twoheadrightarrow_{R\cup S} \cap (A \times A)$ and $\succ = (\to_R \cdot \twoheadrightarrow_{R\cup S}) \cap (A \times A)$.

We verify that $\mathcal{A}$ is an extended well-founded monotone partial Σ-algebra. Suppose $\succ$ would not be well-founded. Then there exists $t \in \mathcal{F}am_{R\cup S}(T)$ admitting an infinite $\to_R \cdot \twoheadrightarrow_{R\cup S}$ rewrite sequence, contradicting $\mathsf{SN}_{R/S}(T)$. We have $\succ \cdot \sqsupseteq \;\subseteq\; \succ$ by definition, and consequently $\mathsf{SN}_{\succ/\sqsupseteq}$ by Lemma 4.4. For $f \in \Sigma$ we show that $[\![f]\!]$ is closed and monotone with respect to $\succ$ (for $\sqsupseteq$ the reasoning is the same). Consider $s, t \in A$ with $s \succ t$. Whenever $[\![f]\!](\dots, s, \dots)\downarrow$ we have also $[\![f]\!](\dots, t, \dots)\downarrow$ (since the family $\mathcal{F}am_{R\cup S}(T)$ is closed under rewriting), and hence $[\![f]\!](\dots, s, \dots) \succ [\![f]\!](\dots, t, \dots)$ as a consequence of the closure of rewriting under contexts. Hence $\mathcal{A}$ is an extended well-founded monotone partial Σ-algebra.

The set T is defined, since for every term $s \in T$ we have $[\![s]\!]\downarrow$ by definition. It remains to be proved that $\succ$ is a partial model for R, and $\sqsupseteq$ a partial model for S. We only consider $\succ$, as the reasoning for $\sqsupseteq$ is the same. Let $\ell \to r \in R$ and $\alpha : \mathcal{X} \to A$ such that $[\![\ell, \alpha]\!]\downarrow$. Then $[\![\ell, \alpha]\!] = \ell\alpha \to_R r\alpha = [\![r, \alpha]\!]$. Then $[\![\ell, \alpha]\!] \succ [\![r, \alpha]\!]$ because both $[\![\ell, \alpha]\!] \in A$ and $[\![r, \alpha]\!] \in A$.

For the 'if'-part assume that $\mathcal{A} = \langle A, [\![\cdot]\!], \succ, \sqsupseteq\rangle$ fulfilling the requirements of the theorem is given. Assume that $\mathsf{SN}_{R/S}(T)$ would not hold. Then there exists $t_0 \in T$ which admits an infinite $\to_R \cup \to_S$ rewrite sequence $t_0 \to t_1 \to \dots$ containing an infinite number of $\to_R$-steps. By Lemma 4.5 this sequence then would give rise to an infinite $\succ \cup \sqsupseteq$ sequence: $[\![t_0]\!]\ (\succ \cup \sqsupseteq)\ [\![t_1]\!]\ (\succ \cup \sqsupseteq) \dots$ containing infinitely many $\succ$-steps, contradicting $\mathsf{SN}_{\succ/\sqsupseteq}$. □

Example 4.7. We consider a simple example to illustrate the method:

$$R = \{a \to b\} \qquad S = \{b \to b,\ f(b) \to f(a)\} \qquad T = \{a\}$$

Global relative termination $\mathsf{SN}_{R/S}$ does not hold, e.g. not on $f(a)$. However on T the rule $a \to b$ is terminating relative to the other rules. We can prove this using the extended well-founded monotone partial Σ-algebra $\mathcal{A} = \langle \{0, 1\}, [\![\cdot]\!], >, \geq\rangle$. The interpretations are given by: $[\![a]\!] = 1$, $[\![b]\!] = 0$ and $[\![f]\!](x)\uparrow$ for all $x \in A$. Then T is defined, $\succ$ is a partial model for R ($[\![a]\!] = 1 > 0 = [\![b]\!]$), and $\sqsupseteq$ is a partial model for S ($[\![b]\!] = 0 \geq 0 = [\![b]\!]$ and $[\![f(b)]\!]\uparrow$). Hence we conclude $\mathsf{SN}_{R/S}(T)$ by an application of Theorem 4.6.

See further Example 9.6 in Section 9 for a non-trivial example.

5. Stepwise Removal of Rules

For termination proofs it is common practice to weaken the proof obligation stepwise by removing rules. The idea is to find interpretations such that a part $R' \subseteq R$ of the rules is decreasing ($\succ$) and the remaining rules are weakly decreasing ($\sqsupseteq$). Then for termination of R it suffices to prove termination of the rules in the complement $R \setminus R'$. We would also like to have this possibility for proofs of local termination. However, for local termination we cannot simply remove (and then forget about) the strictly decreasing rules, as the following example illustrates.

Example 5.1. Consider the set $T = \{a\}$ in the TRS with the following rules:

$$a \to b \qquad\qquad b \to b$$

We define a monotone partial Σ-algebra $\langle \mathbb{N}, [\![\cdot]\!], > \rangle$ by $[\![a]\!] = 1$ and $[\![b]\!] = 0$. Then the rule $a \to b$ is decreasing ($>$ is a partial model) since $[\![a]\!] > [\![b]\!]$, and for $b \to b$ we have $[\![b]\!] = [\![b]\!]$. However, removing the strictly decreasing rule $a \to b$ is not sound, since the resulting TRS is terminating on T.

Let us briefly elaborate on the following theorem which enables us to remove rules stepwise. Assume that the goal is proving that R is terminating relative to S on T, that is, $\mathsf{SN}_{R/S}(T)$. We start with zero knowledge: $\mathsf{SN}_{\varnothing/R\cup S}(T)$. We search for an interpretation that makes a part $R' \subseteq R$ of the rules decreasing ($\succ$) and the remaining rules in $R \cup S$ weakly decreasing ($\sqsupseteq$). Then the rules in R' can only be applied finitely often: $\mathsf{SN}_{R'/((R\setminus R')\cup S)}(T)$. *But how to proceed?* As we have seen above, we cannot simply forget about the rules R', but need to take into account their influence on the family $\mathcal{F}am_{R\cup S}(T)$. A possible and theoretically complete solution would be to require these rules to be weakly decreasing ($\sqsupseteq$). However, for practical applicability this requirement seems too strict as it imposes heavy restrictions on the termination order. We propose a different approach, which allows the 'removed' rules R' to change arbitrarily, even increase, the interpretation of the rewritten terms, as long as rewriting defined terms yields defined terms again. For this purpose we introduce a relation $\rightsquigarrow$ on A, which is a partial model for the already removed rules, and thereby guarantees that these rules preserve definedness.

Theorem 5.2. *Let R, R' and U be TRSs over Σ, and $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$ a set of terms such that $\mathsf{SN}_{U/(R\cup R')}(T)$ holds. Then $\mathsf{SN}_{(U\cup R')/R}(T)$ holds if and only if there exists an extended well-founded monotone partial Σ-algebra $\mathcal{A} = \langle A, [\![\cdot]\!], \succ, \sqsupseteq \rangle$ and a relation $\rightsquigarrow$ on A such that:*

(1) the set T is defined,
(2) $\langle A, [\![\cdot]\!], \rightsquigarrow \rangle$ is a monotone partial Σ-algebra, and
(3) $\succ$, $\sqsupseteq$ and $\rightsquigarrow$ are partial models for R', R and U, respectively.

Proof. Straightforward extension of the proof of Theorem 4.6. The 'only if'-part follows immediately by taking $\rightsquigarrow = \succ$. For the 'if'-part consider an infinite reduction $t_1 \to t_2 \to \dots$ with $t_1 \in T$. Then since $\rightsquigarrow$ is a partial model for U, we conclude $\forall i \in \mathbb{N}.\ [\![t_i]\!]{\downarrow}$. Moreover, as a consequence of $\mathsf{SN}_{U/(R\cup R')}(T)$ we can cut off the prefix of the sequence containing the finitely many U steps. □

Example 5.3. We reconsider Example 3.8, and prove termination of R on T. The usage of Theorem 5.2 allows for a simpler stepwise termination proof. In particular, for removing the rules for *fac* we can employ the standard interpretation *mul* as $\cdot$ and *add* as $+$. Let $\mathcal{A} = \langle \mathbb{N}, [\![\cdot]\!], >, \geq \rangle$ where $>$ is the natural order on $\mathbb{N}$, and $[\![\cdot]\!]$ is given by:

$$[\![0]\!] = 0 \qquad [\![\mathsf{s}]\!](n) = n+1 \qquad [\![\mathit{fac}]\!](n) = (n+2)!$$
$$[\![\mathit{mul}]\!](n,m) = n \cdot m \qquad [\![\mathit{add}]\!](n,m) = n+m \qquad [\![-]\!](n,m)\uparrow$$

for all $n, m \in \mathbb{N}$. Then $>$ is a partial model for (ρ_1) and (ρ_2):

$$[\![\mathit{fac}(0), \alpha]\!] = 2 > 1 = [\![\mathsf{s}(0), \alpha]\!]$$
$$[\![\mathit{fac}(\mathsf{s}(x)), \alpha]\!] = (\alpha(x)+3)! > (\alpha(x)+1)(\alpha(x)+2)! = [\![\mathit{mul}(\mathit{fac}(x), \mathsf{s}(x)), \alpha]\!]$$

and obviously $\geq$ is a partial model for the other rules. Let $U_1 = \{(\rho_1), (\rho_2)\}$, and $R_1 = R \backslash U_1$. Then by Theorem 5.2 it suffices to show $\mathsf{SN}_{U_1/R_1}(T)$ to conclude $\mathsf{SN}_R(T)$.

As second step, we remove the *mul* rules. Let $\mathcal{A} = \langle \mathbb{N}, [\![\cdot]\!], >, \geq \rangle$ with:

$$[\![0]\!] = 0 \qquad [\![\mathsf{s}]\!](n) = n+1 \qquad [\![\mathit{fac}]\!](n) = n$$
$$[\![\mathit{mul}]\!](n,m) = (n+1) \cdot (m+1) \qquad [\![\mathit{add}]\!](n,m) = n+m \qquad [\![-]\!](n,m)\uparrow$$

for all $n, m \in \mathbb{N}$. Recall that the rules from U_1 have to be taken into consideration as they have an impact on the set of reachable terms (otherwise the set of terms T would consist only of normal forms). Nevertheless, the rule (ρ_2) from U_1 is not (weakly) decreasing, that is, $\geq$ is not a partial model for (ρ_2) with respect to the above interpretation:

$$[\![\mathit{fac}(\mathsf{s}(x)), \alpha]\!] = \alpha(x)+1 \not\geq (\alpha(x)+1) \cdot (\alpha(x)+2) = [\![\mathit{mul}(\mathit{fac}(x), \mathsf{s}(x)), \alpha]\!]$$

This is also not necessary. It suffices that U_1 is decreasing with respect to any other relation $\rightsquigarrow$ guaranteeing that all reachable terms are defined. For the current example we can choose the 'total' relation $\rightsquigarrow = \{(n,m) \mid n, m \in \mathbb{N}\}$ relating all pairs of natural numbers. Then $\rightsquigarrow$ is a partial model for U_1, and all $[\![f]\!]$ for $f \in \Sigma$ are closed and monotone with respect to $\rightsquigarrow$. The rules (ρ_4), (ρ_5), and (ρ_6) are decreasing ($>$ is a partial model), for all $\alpha : \mathcal{X} \to \mathbb{N}$:

$$[\![\mathit{mul}(x, 0), \alpha]\!] = \alpha(x)+1 > 0 = [\![0, \alpha]\!]$$
$$[\![\mathit{mul}(0, y), \alpha]\!] = \alpha(y)+1 > 0 = [\![0, \alpha]\!]$$
$$[\![\mathit{mul}(x, \mathsf{s}(y)), \alpha]\!] = (\alpha(x)+1) \cdot (\alpha(y)+2) >$$
$$(\alpha(x)+1) \cdot (\alpha(y)+1) + \alpha(x) = [\![\mathit{add}(\mathit{mul}(x,y), x), \alpha]\!]$$

The remaining rules in R_1 are weakly decreasing (that is, $\geq$ is a partial model). We define $U_2 = U_1 \cup \{(\rho_4), (\rho_5), (\rho_6)\}$, and let $R_2 = R_1 \setminus U_2$. Then by Theorem 5.2 $\mathsf{SN}_{U_2/R_2}(T)$ implies $\mathsf{SN}_{U_1/R_1}(T)$.

Finally, we employ the algebra $\mathcal{A} = \langle \mathbb{N}, [\![\cdot]\!], >, \geq \rangle$ with:

$$[\![0]\!] = 0 \qquad [\![\mathsf{s}]\!](n) = n+1 \qquad [\![\mathit{fac}]\!](n) = n$$
$$[\![\mathit{mul}]\!](n,m) = n+m \qquad [\![\mathit{add}]\!](n,m) = n+2m \qquad [\![-]\!](n,m)\uparrow$$

for all $n, m \in \mathbb{N}$, together with $\rightsquigarrow = \{(n,m) \mid n, m \in \mathbb{N}\}$. Thereby $>$ is a partial model for all rules from R_2, and $\rightsquigarrow$ is a partial model for U_2. Hence, we conclude $\mathsf{SN}_{U_2/R_2}(T)$, and thus $\mathsf{SN}_R(T)$.

For other applications of the theorem see Examples 9.6 and 9.7 in Section 9.

6. Via Models from Local to Global Termination

In this section we describe an easy transformation from local to global termination based on an adaptation of semantic labeling [Zan95]. For this purpose we generalise the concept of models from [Zan95] to partial models. Whenever the language T for which we are interested in termination can be described by a partial model, that is, $T = \{t \mid [\![t]\!]\!\downarrow\}$, then semantic labeling allows for a simple, complete transformation from local to global termination. Here complete means that the original system is locally terminating on T if and only if the transformed, labeled system is globally terminating.

We define a variant of semantic labeling where each symbol is labeled by the tuple of the values of its arguments.

Definition 6.1. Let Σ be a signature, and let $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ be a partial Σ-algebra. For $t \in \mathsf{Ter}(\Sigma, \mathcal{X})$ and $\alpha : Var(t) \to A$ such that $[\![t, \alpha]\!]\!\downarrow$, the *labeling* $lab_{\mathcal{A}}(t, \alpha)$ *of* t *with respect to* α is defined as follows:

$$lab_{\mathcal{A}}(x, \alpha) = x$$
$$lab_{\mathcal{A}}(f(t_1, \ldots, t_n), \alpha) = f^{[\![t_1,\alpha]\!],\ldots,[\![t_n,\alpha]\!]}(lab_{\mathcal{A}}(t_1, \alpha), \ldots, lab_{\mathcal{A}}(t_n, \alpha)) \,.$$

over the signature $lab_{\mathcal{A}}(\Sigma) = \{f^{\lambda} \mid f \in \Sigma,\ \lambda \in A^{\sharp(f)} \text{ such that } [\![f]\!](\lambda)\!\downarrow\}$

In order to obtain a complete transformation we need to restrict the models to their core, that is, those elements that are interpretations of ground terms.

Definition 6.2. Let $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ be a partial Σ-algebra. Then the *core* $\mathcal{A}_c \subseteq A$ *of* $\mathcal{A}$ is the smallest set such that $[\![f]\!](a_1, \ldots, a_n) \in \mathcal{A}_c$ whenever $f \in \Sigma$ and $a_1, \ldots, a_n \in \mathcal{A}_c$ with $[\![f]\!](a_1, \ldots, a_n)\!\downarrow$. We say that $\mathcal{A}$ is *core* if $A = \mathcal{A}_c$.

By construction of the core we have $\mathcal{A}_c = \{[\![t]\!] \mid t \in \mathsf{Ter}(\Sigma, \varnothing),\ [\![t]\!]\!\downarrow\}$. The restriction of a model to its core does not change its language, thus in the sequel we can without loss of generality assume that all models are core.

We have arrived at the transformation from local to global termination. The rules are labeled as known from semantic labeling with the exception that labeled rules are thrown away if the interpretation of their left-hand side is undefined.

Definition 6.3. Let R be a TRS over Σ, and $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ a partial Σ-algebra. We define the *labeling of* R as the TRS $lab_{\mathcal{A}}(R)$ over the signature $lab_{\mathcal{A}}(\Sigma)$ by:

$$lab_{\mathcal{A}}(R) = \{lab_{\mathcal{A}}(\ell, \alpha) \to lab_{\mathcal{A}}(r, \alpha) \mid \ell \to r \in R,\ \alpha : Var(\ell) \to A \text{ such that } [\![\ell, \alpha]\!]\!\downarrow\} \,.$$

A TRS is *collapsing* if it contains rules of the form $\ell \to x$ with $x \in \mathcal{X}$. Such collapsing rules can be eliminated by replacing them with all instances $\ell\sigma_f \to x\sigma_f$ for every $f \in \Sigma$ where $\sigma_f(x) = f(x_1, \ldots, x_n)$ with $x_1, \ldots, x_n$ pairwise different, fresh variables.

Theorem 6.4. *Let R be a non-collapsing TRS over Σ, and $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ a core partial model for R. Then R is locally terminating on $\mathcal{L}(\mathcal{A})$ if and only if $lab_{\mathcal{A}}(R)$ is globally terminating.*

Proof. We introduce types for $lab_{\mathcal{A}}(R)$ over the sorts A. For every symbol $f^{\lambda} \in lab_{\mathcal{A}}(\Sigma)$ with $\lambda = \langle a_1, \ldots, a_{\sharp(f)}\rangle$ we define f^{λ} to have input sorts $\langle a_1, \ldots, a_n\rangle$ and output sort $[\![f]\!](a_1, \ldots, a_n)$. Then [Ohl02, Proposition 5.5.24] with non-collapsingness of $lab_{\mathcal{A}}(R)$ yields that $lab_{\mathcal{A}}(R)$ is terminating if and only if all well-sorted terms are terminating. Since $\mathcal{A}$ is core there exists a well-sorted ground term for every sort in A. Thus by application of a ground substitution we can assume that all rewrite sequences contain only ground terms, and the set of well-sorted ground terms is exactly the language $\mathcal{L}(\mathcal{A})$ of the model $\mathcal{A}$. □

To apply Theorem 6.4 for proving local termination of R on a set of terms T we have to find a partial model $\mathcal{A}$ for R such that $T \subseteq \mathcal{L}(\mathcal{A})$. Then global termination of $lab_{\mathcal{A}}(R)$ implies local termination of R on T. If moreover we have $\mathcal{F}am(T) = \mathcal{L}(\mathcal{A})$, then the transformation is complete, that is, the converse implication holds as well.

Example 6.5. We revisit Example 3.7 on the S combinator with $T = \{\mathsf{S}^n \mid n \in \mathbb{N}\}$. We choose the partial model $\mathcal{A} = \langle A, [\![\cdot]\!] \rangle$, where $A = \{0, 1, 2\}$ and the interpretation is defined by: $[\![\mathsf{S}]\!] = 0$, $[\![@]\!](0,0) = 1$, $[\![@]\!](1,x) = 2$ for all $x \in A$, $[\![@]\!](2,0) = 2$, and $\uparrow$ otherwise. Then $T \subseteq \mathcal{L}(\mathcal{A})$ and a short proof even shows that $\mathcal{F}am(T) = \mathcal{L}(\mathcal{A})$. The labeling $lab_{\mathcal{A}}(\{\mathsf{S}xyz \to xz(yz)\})$ is:

$$@^{2,0}(@^{1,0}(@^{0,0}(\mathsf{S}, x), y), z) \to @^{1,1}(@^{0,0}(x,z), @^{0,0}(y,z))$$
$$@^{2,0}(@^{1,1}(@^{0,0}(\mathsf{S}, x), y), z) \to @^{1,2}(@^{0,0}(x,z), @^{1,0}(y,z))$$
$$@^{2,0}(@^{1,2}(@^{0,0}(\mathsf{S}, x), y), z) \to @^{1,2}(@^{0,0}(x,z), @^{2,0}(y,z))\,.$$

The other labeled rules are thrown out as their left-hand side is undefined. Global termination of the transformed system can be shown by the recursive path order [Der82].

7. Starling and Owl

In this section we consider TRSs with the property that strong and weak normalisation coincide. In case the language of normalising terms happens to be regular, we show how a tree automaton (partial model) can be found accepting exactly the (closed) normalising terms. We automated this procedure. Then we label the TRS with the obtained partial model, and employ Theorem 6.4 to transform the local termination problem for the set of normalising terms to global termination of the labelled TRS.

Since in orthogonal, non-erasing term rewriting systems strong and weak normalisation coincide, these form a typical area where the method can be applied. In this section we illustrate this construction with two well-known examples from combinatory logic (CL) [Cur30]. We use Smullyan's bird nicknames of the combinators [Smu90].

(1) The Owl, corresponding to the rewrite rule:
$$\delta\, xy \to y(xy)$$
(2) The Starling
$$\mathsf{S}\, xyz \to xz(yz),$$
also known as the fragment CL(S) of combinatory logic consisting of all terms solely built from application and the S-combinator.

The termination problem of Smullyan's Owl has been solved in [Klo07]. Here, it serves as illustrating example.

The termination problem of CL(S) is non-trivial, and its word problem is still open. In [Wal00] decidability of strong normalisation of terms in CL(S) has been shown, and we are aiming at a formal verification of the following proposition:

Proposition 7.1 ([Wal00])**.** *The set of normalising ground* S*-terms is a rational language.*

We now turn to the construction of the partial models.

Definition 7.2. For a tree language L, its *Nerode congruence* $\sim_L$ is the relation on ground terms given by $t_1 \sim_L t_2 \iff \forall \text{ground } C[] : C[t_1] \in L \iff C[t_2] \in L$.

The next lemma follows easily by considering the Nerode congruence [CDG+07].

Lemma 7.3. *If a TRS R has the property that every ground term is weakly normalising if and only if it is strongly normalising, and N is the language of normalising ground terms, then*

- *each congruence class of $\sim_N$ is closed under R-rewriting and closed R-expansion,*
- *the complement of N occurs as one of the $\sim_N$ congruence classes.*

Proof. Note that under the assumptions on R, for each term $t \in N$ and each subterm s of a term in N we have $s \in N$. In other words, $s \notin N$ implies $C[s] \notin N$. Also, for all ground terms t_1, t_2 with $t_1 \to_R t_2$ we have $t_1 \in N \iff t_2 \in N$. The claims follow. □

In particular weak and strong normalisation of terms coincide for every orthogonal and non-erasing TRS; this applies for CL(S) as well as Smullyan's Owl. We note that the set of congruence classes of $\sim_N$ can in general be infinite, even for orthogonal, non-erasing TRSs.

Example 7.4. We consider an example of an orthogonal, non-erasing TRS where the set of congruence classes of $\sim_N$ is infinite. Let R consist of the rules:

$$a(b(x)) \to x \qquad\qquad c(c(d)) \to c(c(d))$$

over the signature $\Sigma = \{a, b, c, d\}$ with d a constant. Here, terms of the from $c(a^n(b^n(c(d))))$ are non-terminating, while all terms of the form $c(a^n(b^m(c(d))))$ with $n \neq m$ are terminating. Hence none of the terms $b^n(c(d))$ for $n \in \mathbb{N}$ can be in the same congruence class of $\sim_N$.

Corollary 7.5. *If the set of congruence classes of $\sim_N$ is finite, then the minimal complete deterministic bottom-up tree automaton for N is finite and a model for R.*

Thus, if the set of congruence classes of $\sim_N$ is finite, then the set of normalising terms is a regular language. Assume that we are lucky and the set of congruence classes is finite. *How can we find the regular automaton accepting the set of normalising terms?*

A manual analysis and construction of the automaton as in [Wal00] can be tedious and error-prone. The reference contains a hand-made tree grammar (top-down non-deterministic tree automaton A) and claims:

- the S rule is locally terminating on $L(A)$,
- $L(A)$ contains all normal forms,
- A is closed under inverse application of the S rule.

Starting from that grammar, we can indeed compute a bottom-up minimal deterministic tree automaton B with $L(B) = L(A)$ (strangely, it has 39 states, and not 43, as claimed in the reference).

We propose a different, automatable approach for finding a regular automaton accepting the language of all normalising terms. The idea is to employ the definition of the Nerode congruence for 'guessing' the congruence classes. Here we use the word 'guess' in place of 'compute' since we need to check whether a term $C[s]$ is terminating. This property is in general undecidable. We can, however, make an educated guess by choosing a large enough d and checking whether $C[s]$ admits a rewrite sequence of length d with respect to some strategy $\rightsquigarrow$. Note that the strategy $\rightsquigarrow$ can be chosen arbitrarily since we assume that weak and strong normalisation coincide.

Definition 7.6. [TeR03] A *strategy* $\rightsquigarrow$ for a TRS R is a relation $\rightsquigarrow \subseteq \to_R$ on $\mathsf{Ter}(\Sigma, \mathcal{X})$ having the same normal forms as $\to_R$. A strategy $\rightsquigarrow$ is called *deterministic* if every term t has at most one reduct s, that is, $t \rightsquigarrow s$.

The following algorithm searches for a partial model for the language of normalising terms. The algorithm depends on a strategy $\rightsquigarrow$ for R and parameters $c, d \in \mathbb{N}$ where c is the maximal depth of contexts C, and d is the length of $\rightsquigarrow$-reductions used to guess whether a term is normalising.

Instead of the full Nerode congruence of the set of R-normalizing terms, which might be undecidable, we use an decidable equivalence relation $\sim$ on terms given by $s \sim t$ iff for each context $C[]$ of height $\leq c$, either both $C[s]$ and $C[t]$ have a $\rightsquigarrow$-derivation of length at least d, or both don't.

Algorithm 7.7. Starting from the full relation $\sim_0$ that relates all pairs of terms, we compute successive refinements $\sim_0 \supset \sim_1 \supset \ldots$. Each $\sim_i$ is given as a finite set of representatives $T_i = \{t_{i,1}, \ldots, t_{i,|T_i|}\}$ with $k \neq l \Rightarrow t_{i,k} \not\sim t_{i,l}$, where $\sim$ is the Nerode-like relation defined above. The $\sim_i$-equivalence class of non-normalizing terms is not explicitly represented. By Lemma 7.3, this is no loss of information. The full relation $\sim_0$ is given by $T_0 = \emptyset$. We set $T_{i+1} = T_i \cup \{s\}$ where $s = f(s_1, \ldots, s_a)$, for a choice of function symbol f of arity a, and terms $(s_1, \ldots, s_a) \in T_i^a$, such that the $\rightsquigarrow$-derivation of s has length $< d$, and $s \not\sim t$ for each $t \in T_i$. The algorithm stops if no such s can be found.

We remark that T_1 consists of one element, which is a term containing a nullary symbol only. Each $\sim_i$ computed by this algorithm constitutes a partial algebra (on the carrier T_i), since each step of the algorithm defines one part of the interpretation of a function symbol.

The goal is that the output algebra is a partial model for R, exactly capturing the Nerode congruence. This may fail, for two reasons. If d is chosen too small, then a normalising term $C[s]$ may mistakenly be considered non-normalising. If c is too small, then terms may accidentally be identified, although they behave differently when put into larger contexts.

Nevertheless, it can be shown that if the language of normalising terms is regular, then there exist appropriate parameters c and d such that the algorithm will compute the correct partial model, see Lemma 7.8. The case of having chosen c or d too small can be detected after running the algorithm as follows. Let $\mathcal{A}$ be the algebra computed by the algorithm. It can be effectively checked whether $\mathcal{A}$ is a partial model for R, and whether all undefined terms $[\![t]\!]\uparrow$ contain a redex with respect to R. Then it automatically follows that all undefined terms are non-normalising. Finally we can employ Theorem 6.4 to transform the termination problem for all defined terms $[\![t]\!]\downarrow$ into an equivalent global termination problem of $lab_{\mathcal{A}}(R)$. If we find a termination proof for $lab_{\mathcal{A}}(R)$, then the partial model $\mathcal{A}$ is correct and accepts exactly the language of normalising terms.

Lemma 7.8. *Let R be a TRS such that every ground term is weakly normalising if and only if it is strongly normalising. If the language N of normalising terms is regular, then there exist appropriate parameters c and d such that Algorithm 7.7 computes the correct partial model accepting exactly N.*

Proof. If the languages N is regular, then the set of congruence classes of $\sim_N$ is finite. Let $n = |\mathsf{Ter}(\Sigma, \varnothing)/_{\sim_N}|$, and let $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$ be the set of all ground terms of height $\leq n+1$. Then T contains at least one representative $t_D \in T$ for every $D \in \mathsf{Ter}(\Sigma, \varnothing)/_{\sim_N}$. For every pair $\langle t_{D_1}, t_{D_2} \rangle$ of representatives with $t_{D_1} \neq t_{D_2}$ we pick a 'discriminating' context C such

that $C[t_{D_1}] \in N \not\Leftrightarrow C[t_{D_2}] \in N$. Let $\mathcal{C}$ be the set of these (finitely many) contexts. We choose for c the maximal depth of all contexts in $C \in \mathcal{C}$, and for d the maximal length of a $\rightsquigarrow$-reduction of all normalising $C[t] \in N$ with $C \in \mathcal{C}$ and $t \in T$. Then the choice of d guarantees that terms $C[t]$ will not accidentally be identified as non-terminating, and the choice of c guarantees that all non-equivalent terms in T_i will be distinguished. □

We have implemented Algorithm 7.7; the Haskell source can be downloaded from:

`http://infinity.few.vu.nl/local/`

We have applied the algorithm on Smullyan's Owl and CL(S), obtaining in both cases the minimal partial algebra accepting the language of all normalising terms. Further details, including the respective partial models, are given below.

Example 7.9 (**Smullyan's Owl**)**.** Smullyan's Owl serves as illustrating example. The set of normalising Owl-terms has been found in [Klo07]. The Owl corresponds to the following rewrite rule:

$$\delta\ xy \to y(xy)$$

or, equivalently, in first order notation:

$$@(@(\delta, x), y) \to @(y, @(x, y)) \tag{7.1}$$

Applied to Rule (7.1), Algorithm 7.7 computes the partial model $\mathcal{A} = \langle \{0, 1\}, [\![\cdot]\!] \rangle$ where $[\![\delta]\!] = 0$, and the interpretation of @ is given in Table 1.

	0	1
0	1	1
1	1	-

Table 1: Interpretation $[\![@]\!](x, y)$ for the Owl with x on the left and y on the top.

Examples for terms $t \in \mathcal{L}(\mathcal{A})$, that is, normalising terms, are

$$\delta\delta\delta \ldots \delta, \qquad \delta(\delta\delta \ldots \delta)\delta \ldots \delta, \text{ and} \qquad \delta(\delta(\delta\delta)\delta\delta)\delta\delta\delta$$

For an example of a non-normalising term take $\delta\delta(\delta\delta)$. In words, the set of undefined (non-normalising) terms can be described as follows: a term is undefined if it contains two distinct occurrences of $\delta\delta$. Note that the term $\delta\delta\delta \ldots \delta = (\ldots((\delta\delta)\delta)\ldots)\delta$ contains only one occurrence of $\delta\delta$ (or $@(\delta, \delta)$ in first-order notation).

First, we check that $\mathcal{A}$ is a partial model for R:

$$\begin{aligned}
[\![\delta\ xy, \alpha]\!] = 1 = [\![y(xy), \alpha]\!] &\qquad \text{for } \alpha(x) = 0,\ \alpha(y) = 0 \\
[\![\delta\ xy, \alpha]\!]{\uparrow} \text{ and } [\![y(xy), \alpha]\!]{\uparrow} &\qquad \text{for } \alpha(x) = 0,\ \alpha(y) = 1 \\
[\![\delta\ xy, \alpha]\!] = 1 = [\![y(xy), \alpha]\!] &\qquad \text{for } \alpha(x) = 1,\ \alpha(y) = 0 \\
[\![\delta\ xy, \alpha]\!]{\uparrow} \text{ and } [\![y(xy), \alpha]\!]{\uparrow} &\qquad \text{for } \alpha(x) = 1,\ \alpha(y) = 1
\end{aligned}$$

Second, we use induction on the term structure to show that every undefined term contains a redex. Let $t \in \mathsf{Ter}(\Sigma, \varnothing)$ be a term such that $[\![t]\!]{\uparrow}$. Then by definition of $[\![\cdot]\!]$ the term t is of the form $t = @(t_1, t_2)$ and either $[\![t_1]\!] = [\![t_2]\!] = 1$, or $[\![t_1]\!]{\uparrow}$, or $[\![t_2]\!]{\uparrow}$. In the latter two cases it suffices to apply the induction hypothesis to t_1 or t_2, respectively. Thus, let $[\![t_1]\!] = [\![t_2]\!] = 1$. We use induction on the term structure of t_1. Again, by definition of $[\![\cdot]\!]$ the term t_1 is of the form $t_1 = @(t_1', t_2')$ with $[\![t_1']\!] = 0$, or $[\![t_1']\!] = 1$. If $[\![t_1']\!] = 0$, then $t_1' = \delta$

and $t = @(@(\delta, t_2'), t_2)$, and hence t contains a redex. For $[\![t_1']\!] = 1$ we finish by applying the second induction hypothesis.

Third, we prove termination for all defined terms. An application of Theorem 6.4 yields the following labelled TRS:

$$@1,0(@0,0(\delta,x),y) \to @0,1(y,@0,0(x,y)) \qquad \text{for } \alpha(x)=0,\ \alpha(y)=0$$
$$@1,0(@0,1(\delta,x),y) \to @0,1(y,@1,0(x,y)) \qquad \text{for } \alpha(x)=1,\ \alpha(y)=0$$

Termination of this system can easily be proven; for example AProVE [GTSKF04] finds a termination proof using the recursive path order.

Thus, indeed, $\mathcal{A}$ is a partial model accepting exactly the normalising δ-terms.

Example 7.10 (**The set of normalising S-terms**)**.** For CL(S), Algorithm 7.7 returns the partial model $\mathcal{A} = \langle A, [\![\cdot]\!] \rangle$ where $A = \{0, 1, \ldots, 37\}$, $[\![\mathsf{S}]\!] = 4$, and the interpretation of $@$ is given in Table 2. Indeed, it can be checked that $\mathcal{A}$ is equivalent to the grammar given in [Wal00].

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
0	36	36	36	27	1	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
1	34	35	35	32	2	36	37	37	37	36	37	36	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
2	36	36	36	33	2	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
3	2	2	2	1	0	9	8	10	8	9	10	12	12	14	14	16	16	24	18	19	20	21	22	26	24	25	26	27	28	29	30	31	32	33	34	35	36	37
4	6	7	7	5	3	11	18	18	18	13	18	15	15	18	18	18	18	28	28	23	26	26	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28
5	32	32	32	17	19	27	33	33	33	33	33	33	33	33	33	33	33	33	33	33	31	31	33	33	33	33	33	33	33	31	31	31	33	33	36	36	36	37
6	34	34	34	29	20	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	37
7	35	35	35	30	21	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	36	37
8	36	36	36	31	20	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
9	36	36	36	33	19	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
10	36	36	36	31	21	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
11	36	36	36	22	25	36	37	37	37	36	37	36	36	37	37	37	37	37	37	36	36	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
12	36	36	36	33	25	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
13	36	36	36	31	30	36	37	37	37	36	37	36	36	37	37	37	37	37	37	36	36	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
14	36	36	36	31	30	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
15	36	36	36	31	31	36	37	37	37	36	37	36	36	37	37	37	37	37	37	36	36	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
16	36	36	36	31	31	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
17	36	36	36	35	36	37	37	37	37	37	37	37	37	37	37	37	37		37	37				37		37	37		37										
18	37	37	37	37	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
19	37	37	37	36	27	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37			37	37	37	37	37	37	37				37	37				
20				37	32																																	
21				37	33																																	
22	37	37	37	36	37	37	37	37	37	37	37	37	37	37	37	37	37		37					37			37		37									
23	36	36	36	36	36	36	37	37	37	36	37	36	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
24	36	36	36	36	36	37	37	37	37	37	37	37	37	37	37	37	37		37	37				37		37	37		37									
25	37	37	37	37	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37			37	37	37	37	37	37	37				37	37				
26	36	36	36	36	36	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
27	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37		37	37				37		37	37		37									
28	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37
29				37	34																																	
30				37	36																																	
31				37	37																																	
32	37	37	37	37	36	37	37	37	37	37	37	37	37	37	37	37	37		37					37			37		37									
33	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37	37		37					37			37		37									
34					35																																	
35					36																																	
36					37																																	
37																																						

Table 2: Transition table for $[\![@]\!](x, y)$ with x left, y top.

We have formally verified (using the proof assistant Coq [Coq]) that $\mathcal{A}$ is a partial model for CL(S) and that the language of $\mathcal{A}$ contains all normalising terms.

For proving that CL(S) is terminating on the language of $\mathcal{A}$ we have transformed the local into a global termination problem using Definition 6.3. The resulting TRS contains 1800 rules which are globally terminating, as can be shown using the DP transformation with SCC decomposition [AG00] together with simple projections and the subterm criterion [HM04]. The termination proof can be found automatically, and formally verified using the current versions of CeTA (1.05) [TS09] and TTT2 (1.0) [KSZM09].

8. The RFC Method

We show that the method proposed in Section 6 is not only useful for local termination, but can fruitfully be employed for global termination as well. In [Der81], Dershowitz reduces global termination of right-linear TRSs to local termination on the set $\mathrm{RFC}(R)$, called the right-hand sides of forward closures of R. The set $\mathrm{RFC}(R) \subseteq \mathsf{Ter}(\Sigma, \mathcal{X})$ is a subset of all terms, weakening the proof obligation, and often allowing for simpler termination proofs. Previously, the only automated method employing this transformation for proving global termination has been the method of match-bounded string rewriting [GHW04]. In the present paper we advocate an alternative approach.

We propose a combination of the RFC-method with the transformation from Section 6. More precisely, we first reduce the global termination problem to a local termination problem on $\mathrm{RFC}(R)$, and then we transform this problem back into a global termination problem. We show that this method can successfully be applied to obtain proofs for global termination; see further Example 8.3 for a rewrite system that remained unsolved in the termination competition [Ter08].

A string rewriting systems (SRS) R is a TRS R where all symbols $f \in \Sigma$ of the signature are unary. We then use words $a_1 \dots a_n$ to denote terms $a_n(\dots a_1(x))$.

For SRSs R the set $\mathrm{RFC}(R)$ can be defined as follows:

Definition 8.1 ([Der81])**.** Let R be a SRS over Σ. The *right-hand sides of forward closures of* R, denoted $\mathrm{RFC}(R)$, are defined as the smallest set $F \subseteq \Sigma^*$ such that:

- $\mathrm{rhs}(R) \subseteq F$,
- if $u \in F$ and $u \to v$, then $v \in F$ (rewriting), and
- if $u\ell_1 \in F$ and $\ell_1\ell_2 \to r \in R$ with $\ell_1 \neq \varepsilon$, then $ur \in F$ (right extension).

We have the following well-known theorem:

Theorem 8.2 ([Der81])**.** *A string rewriting system R is terminating on Σ^* if and only if R is terminating on* $\mathrm{RFC}(R)$.

The set $\mathrm{RFC}(R)$ can be (over-)approximated by using the system

$$R_{\#} = \{u\# \to r\# \mid (u \cdot v \to r) \in R, u \neq \epsilon, v \neq \epsilon\}$$

over $\Sigma_{\#} = \Sigma \cup \{\#\}$, where $\#$ acts as an end marker. Then $\mathrm{RFC}(R)\# = (R \cup R_{\#})^*(\mathrm{rhs}(R)\#)$, and we can reduce the global termination problem of R to local termination of $R \cup R_{\#}$ on $\mathrm{rhs}(R)\#$. More generally we have the following observation: whenever $M \supseteq \mathrm{rhs}(R)\#$ is closed w.r.t. $R \cup R_{\#}$, then $\mathrm{RFC}(M)\# \subseteq M$. The closure under rewriting can be proven by giving a partial model $\mathcal{A}$ (M is the language of a partial $\Sigma_{\#}$-algebra $\mathcal{A}$).

Example 8.3. Take $\Sigma = \{a, b, c\}$ and

$$R = \{a \to \epsilon, b \to \epsilon, cc \to a, ba \to cacbb\}.$$

This is the mirrored version of `SRS/Waldmann07b/size-12-alpha-3-num-223` which has not been solved automatically in previous termination competitions. We present a partial $\Sigma_{\#}$-algebra $\mathcal{A}$ with 3 elements $A = \{1, 2, 3\}$ and interpretations of function symbols:

$$a : 1 \mapsto 1, 2 \mapsto 2; b : 1 \mapsto 1, c : 1 \mapsto 2, 2 \mapsto 1, \# : 1 \mapsto 3, 2 \mapsto 3,$$

and $b(2)$ as well as all transitions from 3 are undefined. Note: we consider the right end of the string to be the top symbol of the term. It can be checked that $\mathcal{A}$ is a partial model for $R \cup R_{\#}$, and its language contains $\mathrm{rhs}(R)\#$. As a consequence we have $\mathcal{L}(\mathcal{A}) \subseteq \mathrm{RFC}(R)\#$.

Formally, for the existence of ground terms, we add a fresh constant e with interpretation $[\![e]\!] = 1$. This constant does not harm the property of $\mathcal{A}$ being a partial model for $R \cup R_\#$, and does not affect the termination behaviour: since SRSs are linear, $R \cup R_\#$ is terminating on $\mathrm{rhs}(R)\#$ if and only if $R \cup R_\#$ is terminating on $\{a_n(\dots a_1(e)) \mid a_1 \dots a_n \in \mathrm{rhs}(R)\#\}$.

We obtain the following labelled system:

$$R_\mathcal{A} = \{a_1 \to \epsilon, a_2 \to \epsilon, b_1 \to \epsilon, c_1c_2 \to a_1, c_2c_1 \to a_2, b_1a_1 \to c_1a_2c_2b_1b_1\},$$

termination of which is equivalent to termination of R. Indeed $R_\mathcal{A}$ is easily seen to be terminating. E.g., Torpa [Zan05] finds the following termination proof:

```
[A] Choose polynomial interpretation
      a1 c1: lambda x.x+1,
      rest identity
    remove: a1  ->
    remove: c2 c1  -> a2
[AC] Reverse every lhs and rhs and choose polynomial
     interpretation:
      a1 and c1: lambda x.10x,
      rest lambda x.x+1
    remove: a2  ->
    remove: b1 a1  -> c1 a2 c2 b1 b1
    remove: b1  ->
    remove: c1 c2  -> a1
Terminating since no rules remain.
```

For automating this method, the challenge is to find a partial model such that the resulting labelled total termination problem is easier than the original one. In particular, the domain of the partial algebra must be a proper subset of the full algebra (Σ^*). In our example, the domain excludes all words containing the factor *bcb*. □

9. Monotone-models for Local Termination

In Sections 3–5 we have devised a characterisation of local termination in terms of monotone partial algebras. While this gives the general method, for the purpose of obtaining automatable methods we strive for fruitful classes of these algebras. For global termination, instances of monotone algebras are well-known. This raises the natural question whether we can transform a given monotone algebra for global termination in such a way that we obtain a partial monotone algebra for local termination.

In this section we present one such approach. We combine monotone partial models with (ordinary) monotone algebras. The monotone partial models are roughly deterministic tree automata that are closed under rewriting; they describe the language of term on which we proof termination. We search for such an automaton that accepts the starting language T together with a monotone algebra such that the rewrite rules decrease on the language of the automaton. In this way monotone algebras for global termination carry over to local termination, and we obtain an automatable method that is applicable for proofs of local termination.

First we give the definition of extended μ-monotone algebras as known from global termination of context-sensitive TRSs, see [Luc98, EWZ08]. A mapping $\mu : \Sigma \to 2^\mathbb{N}$ is

called a *replacement map (for Σ)* if for all $f \in \Sigma$ we have $\mu(f) \subseteq \{1, \ldots, \sharp(f)\}$. Let $\langle A, [\![\cdot]\!]\rangle$ be a Σ-algebra and μ a replacement map. For symbols $f \in \Sigma$ we say that the interpretation $[\![f]\!] : A^{\sharp(f)} \to A$ is *μ-monotone* with respect to $\succ$ if for every $a, b \in A$ and $i \in \mu(f)$ with $a \succ b$ we have: $f(\underbrace{\ldots}_{i-1}, a, \underbrace{\ldots}_{\sharp(f)-i}) \succ f(\ldots, b, \ldots)$.

Definition 9.1. Let μ be a replacement map for Σ.
An *extended well-founded μ-monotone Σ-algebra* $\langle A, [\![\cdot]\!], \succ, \sqsupseteq\rangle$ is a Σ-algebra $\langle A, [\![\cdot]\!]\rangle$ with two binary relations $\succ$, $\sqsupseteq$ on A for which the following conditions hold:

(i) $\mathsf{SN}_{\succ/\sqsupseteq}$, and
(ii) for every $f \in \Sigma$ the function $[\![f]\!]$ is μ-monotone with respect to $\succ$ and $\sqsupseteq$.

A partial model $\mathcal{A} = \langle A, [\![\cdot]\!], \geq\rangle$ may contain elements $a \in A$ for which $[\![t]\!] = a$ implies that t is a normal form. For a given partial model the set of these, which we denote by $A_{nf(R)}$, can be computed (see below Definition 9.2). We can exploit this knowledge as follows: if a certain argument of a symbol $f \in \Sigma$ is always a normal form, then its interpretation $[\![f]\!]$ does not need to be monotonic for this argument position. The following definition gives an algorithm for computing the set $A_{nf(R)}$. Elements that are interpretations $[\![\ell, \alpha]\!]$ of left-hand sides in R cannot belong to this set. Moreover if $a \notin A_{nf(R)}$ and $b = [\![f]\!](\ldots, a, \ldots)$ then we conclude $b \notin A_{nf(R)}$. This is formalised as follows:

Definition 9.2. Let R be a TRS over the signature Σ, and $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ a partial Σ-algebra. The *normal forms $A_{nf(R)}$ of $\mathcal{A}$* are the largest set $A_{nf(R)} \subseteq \mathcal{A}_c$ such that $[\![\ell, \alpha]\!] \notin A_{nf(R)}$ for every $\ell \to r \in R$ and every $\alpha : Var(\ell) \to \mathcal{A}_c$, and $[\![f]\!](a_1, \ldots, a_n) \notin A_{nf(R)}$ for every $f \in \Sigma$, $a_i \notin A_{nf(R)}$ and $a_1, \ldots, a_n \in \mathcal{A}_c$.

Then by construction we obtain the following lemma:

Lemma 9.3. *$A_{nf(R)}$ consists of all $a \in \mathcal{A}_c$ for which every term $t \in \mathsf{Ter}(\Sigma, \varnothing)$ with $[\![t]\!] = a$ is a normal form with respect to R.*

As mentioned above the interpretations do not need to be monotonic in argument positions which are normal forms. We formalise this by defining a replacement map for the labelling $lab_{\mathcal{A}}(R)$ of R which does not contain argument positions that are in normal form.

Definition 9.4. Let R be a TRS over Σ, and $\mathcal{A} = \langle A, [\![\cdot]\!]\rangle$ a partial Σ-algebra. Let the replacement map $\mu^{nf(R)}$ be defined for every symbol $f^\lambda \in lab_{\mathcal{A}}(\Sigma)$ with $\lambda = \langle a_1, \ldots, a_{\sharp(f)}\rangle$ as follows: $\mu^{nf(R)}(f^\lambda) = \{1, \ldots, \sharp(f)\} \setminus \{i \mid a_i \in A_{nf(R)}\}$.

As an instance of Theorem 5.2 we obtain a method for stepwise rule removal for local termination that is based on a combination of monotone partial models and extended monotone algebras.

Theorem 9.5. *Let R, R' and U be TRSs over Σ, and $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$ a set of terms such that $\mathsf{SN}_{U/R\cup R'}(T)$ holds. Furthermore let $\mathcal{A} = \langle A, [\![\cdot]\!], \geq\rangle$ be a monotone partial model for $R \cup R' \cup U$ with $T \subseteq \mathcal{L}(\mathcal{A})$, and $\mathcal{B} = \langle B, [\![\cdot]\!]_{\mathcal{B}}, \succ, \sqsupseteq\rangle$ an extended well-founded $\mu^{nf(R\cup R')}$-monotone $(lab_{\mathcal{A}}(\Sigma))$-algebra such that:*
(1) *$\langle B, \succ\rangle$ is a model for $lab_{\mathcal{A}}(R')$,*
(2) *$\langle B, \sqsupseteq\rangle$ is a model for $lab_{\mathcal{A}}(R)$, and*
(3) *for all $f \in \Sigma$, $\vec{a}_1\, a\, \vec{a}_2 \in A^{\sharp(f)}$, $a \geq a' \in A$, and $b_1, \ldots, b_{\sharp(f)} \in B$:*

$$[\![f^{\vec{a}_1\, a\, \vec{a}_2}]\!]_{\mathcal{B}}(b_1, \ldots, b_{\sharp(f)}) \sqsupseteq [\![f^{\vec{a}_1\, a'\, \vec{a}_2}]\!]_{\mathcal{B}}(b_1, \ldots, b_{\sharp(f)}) \,.$$

Then $\mathsf{SN}_{(U\cup R')/R}(T)$ *holds.*

Proof. We construct an extended well-founded monotone partial Σ-algebra $\mathcal{C} = \langle C, [\![\cdot]\!]_{\mathcal{C}}, \succ_{\mathcal{C}}, \sqsupseteq_{\mathcal{C}}\rangle$ fulfilling the requirements of Theorem 5.2. Let $C = A \times B$, and define $\langle a_1, b_1\rangle \succ_{\mathcal{C}} \langle a_2, b_2\rangle \iff a_1 \notin A_{nf(R\cup R')}$ & $a_1 \geq a_2$ & $b_1 \succ b_2$ and $\langle a_1, b_1\rangle \sqsupseteq_{\mathcal{C}} \langle a_2, b_2\rangle \iff a_1 \notin A_{nf(R\cup R')}$ & $a_1 \geq a_2$ & $b_1 \sqsupseteq b_2$. Note that the $\mu^{nf(R\cup R')}$-monotonicity is implemented by excluding elements $\langle a_1, b_1\rangle$ with $a_1 \in A_{nf(R\cup R')}$ from being sources of $\succ \cup \sqsupseteq$ steps. Then for each $f \in \Sigma$: $[\![f]\!]_{\mathcal{C}}(\langle a_1, b_1\rangle, \ldots, \langle a_{\sharp(f)}, b_{\sharp(f)}\rangle) = \langle [\![f]\!]_{\mathcal{A}}(a_1, \ldots, a_{\sharp(f)}), [\![f^{a_1,\ldots,a_{\sharp(f)}}]\!]_{\mathcal{B}}(b_1, \ldots, b_{\sharp(f)})\rangle$ if $[\![f]\!]_{\mathcal{A}}(a_1, \ldots, a_{\sharp(f)})\downarrow$, and $\uparrow$ otherwise. Finally, we define the relation $\rightsquigarrow$ on C by $\langle a_1, b_1\rangle \rightsquigarrow \langle a_2, b_2\rangle \iff a_1 \geq a_2$. Now it is straightforward to check that all requirements of Theorem 5.2 are fulfilled, and we conclude $\mathsf{SN}_{(U\cup R')/R}(T)$. □

Let us briefly elaborate on the theorem. As an instance of Theorem 5.2, Theorem 9.5 is applicable for proving local termination as well as local relative termination. We start without knowledge $\mathsf{SN}_{\varnothing/R\cup S}(T)$ and stepwise 'remove' rules, more precisely, we move rules from the right side to the left side of the slash '/'. If we reach the goal $\mathsf{SN}_{R/S}(T)$, then the proof has been successful.

The use of partial monotone partial models for $R \cup R' \cup U$ with $T \subseteq \mathcal{L}(\mathcal{A})$ guarantees that the language we consider is closed under rewriting. The set R' is the set of strictly decreasing rules that we are aiming to remove. The $\mu^{nf(R\cup R')}$-monotone $lab_{\mathcal{A}}(\Sigma)$-algebra $\mathcal{B}$ then has the task to make all labelled rules stemming from R' strictly decreasing ($\succ$), and from R weakly decreasing ($\sqsupseteq$). Then we conclude that $R' \cup U$ is terminating relative to R on T.

Example 9.6 (Klop, see [Bar84], Exercise 7.4.7)**.** Example 3.7 can be generalised to include the combinator K, which has the reduction rule $\mathsf{K}xy \to x$. The initial language of flat S,K-terms is $T = (\mathsf{S}|\mathsf{K})^*$; for example $\mathsf{SSKS} = (((\mathsf{SS})\mathsf{K})\mathsf{S})$. The partial model presented in Example 6.5 can be extended to a monotone partial model for this generalised example by fixing $[\![\mathsf{K}]\!] = 0$ and $2 > 0$, $2 > 1$. Note that this is not a model due to $[\![\mathsf{K}xy, \alpha]\!] = 2 > 0 = [\![x, \alpha]\!]$ for $\alpha = \lambda z.0$. For the complete proof, employing this model, we refer to:

`http://infinity.few.vu.nl/local/`.

The second example illustrates the stepwise rule removal.

Example 9.7. We use a Turing-machine-like TRS which does the following. Starting with its head between two symbols 1, the tape containing a finite string of 1's and further blanks (0), it initially puts two boxes □ left and right of its head and afterwards alternately runs left and right between the boxes, each time moving them one position further, until the blanks are reached:

$$\begin{aligned} &11\Box 1R11111\Box 11 \twoheadrightarrow 11\Box 111111R\Box 11\\ \to\ &11\Box 111111L1\Box 1 \twoheadrightarrow 11\Box L1111111\Box 1\\ \to\ &1\Box 1R1111111\Box 1 \twoheadrightarrow \ldots \end{aligned}$$

This is implemented by the TRS R consisting of the following rules:

$$\begin{array}{llll} 1S1 \to \Box R\Box & R1 \to 1R & R\Box 1 \to L1\Box & R\Box 0 \to F\Box 0\\ & 1L \to L1 & 1\Box L \to \Box 1R & 0\Box L \to 0\Box R\\ & 1F \to F1 & 1\Box F \to \Box 1R & 0\Box F \to \mathit{finish} \end{array}$$

where all symbols apart form *finish* (which is a constant) are unary, but have been written without parenthesis for the purpose of compactness. Note that the construction of the TRS is similar to the standard translation of Turing machines to string rewriting systems as given in [TeR03].

While the Turing machine is terminating on every input, the TRS R fails to be globally terminating. The reason is that R allows for configurations with multiple heads working at the same time on the same tape:

$$0\Box R\Box 1F\Box 0 \to 0\Box L1\Box F\Box 0 \to 0\Box L\Box 1R\Box 0 \to^2 0\Box R\Box 1F\Box 0 \to \dots$$

We will prove that R is locally terminating on all terms containing arbitrary occurrences of the symbols 0, 1 and at most one occurrence of S, that is, the language given by $T = \{0,1\}^*\, S\, \{0,1\}^*\mathit{finish}$. As the first step we remove the rules $1S1 \to \Box R\Box$ and $0\Box F \to \mathit{finish}$. We do this by using a monotone model $\mathcal{A}$ consisting of only one element, accepting all terms. We combine this model with the $lab_{\mathcal{A}}(\Sigma)$-algebra $\mathcal{B}$ where $B = \mathbb{N}$ and $[\![1]\!]_{\mathcal{B}}(x) = [\![0]\!]_{\mathcal{B}}(x) = x+1$, all other symbols are interpreted as $\lambda x.x$. This makes the above two rules decreasing ($\succ$ is a model for them).

In the second step, we use a partial model $\mathcal{A} = \langle A, [\![\cdot]\!], \geq\rangle$ where $A = \{0,1\}$, $0 \geq 0$, $1 \geq 1$ (but not $1 \geq 0$), $[\![\mathit{finish}]\!] = 0$ and the other interpretations are given in Table 3:

x	$[\![1]\!](x)$	$[\![\Box]\!](x)$	$[\![R]\!](x)$	$[\![L]\!](x)$	$[\![F]\!](x)$	$[\![0]\!](x)$	$[\![S]\!](x)$
0	0	1	$\uparrow$	$\uparrow$	$\uparrow$	0	0
1	1	0	1	1	1	$\uparrow$	$\uparrow$

Table 3: Symbol interpretations.

As required by the theorem $\mathcal{A}$ is a monotone partial model for R including the two removed rules $U = \{1S1 \to \Box R\Box,\ 0\Box F \to \mathit{finish}\}$ (without them T would consist of normal forms). We use this partial model together with the extended well-founded monotone $lab_{\mathcal{A}}(\Sigma)$-algebra $\mathcal{B} = \langle \mathbb{N}, [\![\cdot]\!]_{\mathcal{B}}, \succ, \sqsupseteq\rangle$ where $\succ$ and $\sqsupseteq$ are the usual orders $>$ and $\geq$ on $\mathbb{N}$, respectively. The interpretation $[\![\cdot]\!]_{\mathcal{B}}$ is $[\![\mathit{finish}]\!]_{\mathcal{B}} = 7$, $[\![1^0]\!]_{\mathcal{B}}(x) = 2\cdot x + 1$, $[\![1^1]\!]_{\mathcal{B}}(x) = 2\cdot x$, $[\![\Box^0]\!]_{\mathcal{B}}(x) = [\![\Box^1]\!]_{\mathcal{B}}(x) = x$, $[\![R^1]\!]_{\mathcal{B}}(x) = 2\cdot x$, $[\![L^1]\!]_{\mathcal{B}}(x) = 2\cdot x + 1$, $[\![F^1]\!]_{\mathcal{B}}(x) = 2\cdot x$, $[\![0^0]\!]_{\mathcal{B}}(x) = 2\cdot x$, and $[\![S^0]\!]_{\mathcal{B}}(x) = 5\cdot x + 6$. Then R' consists of the following rules: $R\Box 1 \to L1\Box$, $1L \to L1$, $1\Box L \to \Box 1R$, $0\Box L \to 0\Box R$, and $1\Box F \to \Box 1R$. Then $\langle \mathcal{B}, \succ\rangle$ is a model for $lab_{\mathcal{A}}(R')$. For instance consider the rule $R\Box 1 \to L1\Box$. The labelling $R^1\Box^0 1^0 \to L^1 1^1 \Box^0$ is in $lab_{\mathcal{A}}(R')$ and its interpretation in $\mathcal{B}$ is: $R^1\Box^0 1^0(x) = 4\cdot x + 2 > 4\cdot x + 1 = L^1 1^1 \Box^0(x)$. The labelling $R^0\Box^1 1^1 \to L^0 1^0 \Box^1$ is not in $lab_{\mathcal{A}}(R')$ since its left-hand side is undefined with respect to $\mathcal{A}$, thus we can ignore this rule. Analogously it can be verified $\langle \mathcal{B}, \sqsupseteq\rangle$ is a model for $lab_{\mathcal{A}}(R \backslash R')$. Since $>$ is the empty relation on A the third condition of Theorem 9.5 holds trivially.

The three remaining rules $R1 \to 1R$, $1F \to F1$, and $R\Box 0 \to F\Box 0$ are even globally terminating. This corresponds to taking a model which has only one state and accepts all terms together with the corresponding termination order which proves global termination. Hence we have proven $\mathsf{SN}_R(T)$ by three consecutive applications of Theorem 9.5.

Finally, we give a theorem that allows us to remove rules and forget about them. We need to be sure that these rules do not influence the family, that is, the set of reachable terms. This is guaranteed if all terms in the family are normal forms with respect to these rules.

Theorem 9.8. *Let R, R' and S be TRSs over Σ, and $T \subseteq \mathsf{Ter}(\Sigma, \varnothing)$. Let $\mathcal{A} = \langle A, [\![\cdot]\!], \geq \rangle$ be a monotone partial model for $R \cup R' \cup S$ with $T \subseteq \mathcal{L}(\mathcal{A})$ such that for all rules $\ell \to r \in R'$ and $\alpha : Var(\ell) \to A$ we have $[\![\ell, \alpha]\!]\uparrow$ (the left-hand side is undefined). Then $\mathsf{SN}_{R/S}(T)$ implies $\mathsf{SN}_{R \cup R'/S}(T)$.*

Proof. From $\mathcal{F}am_{R \cup R' \cup S}(T) \subseteq \mathcal{L}(\mathcal{A})$ together with $[\![\ell, \alpha]\!]\uparrow$ for all $\ell \to r \in R'$ and α it follows that the rules in R' are not reachable. All terms in $\mathcal{F}am(T)$ are normal forms with respect to R'. Hence we can ignore these rules. □

Example 9.9. Consider the TRS R consisting of the following four rules:

$$f(\mathsf{s}(\mathsf{s}(x))) \to f(o(x)) \qquad o(\mathsf{s}(\mathsf{s}(x))) \to \mathsf{s}(\mathsf{s}(o(x))) \qquad o(0) \to 0 \qquad o(\mathsf{s}(0)) \to \mathsf{s}(\mathsf{s}(0)))$$

The TRS is not terminating: $f(\mathsf{s}(\mathsf{s}(\mathsf{s}(0)))) \to f(o(\mathsf{s}(0))) \to f(\mathsf{s}(\mathsf{s}(\mathsf{s}(0)))) \to \ldots$. However, the function f is terminating when applied to an even number, that is, the language $T = \{f(\mathsf{s}^{2 \cdot n}(0)) \mid n \in \mathbb{N}\}$. We choose $\mathcal{A} = \langle \{0, 1\}, [\![\cdot]\!], \geq \rangle$ where $[\![0]\!] = 0$, $[\![\mathsf{s}]\!](0) = 1$, $[\![\mathsf{s}]\!](1) = 0$, $[\![o]\!](0) = 0$, $[\![o]\!](1)\uparrow$, $[\![f]\!](0) = 0$ and $[\![f]\!](1)\uparrow$. Then $\mathcal{A}$ is a monotone partial model with $T \subseteq \mathcal{L}(\mathcal{A})$. We have $[\![o(\mathsf{s}(0)), \alpha]\!]\uparrow$ (for all α), thus the rule $o(\mathsf{s}(0)) \to \mathsf{s}(\mathsf{s}(0)))$ is never applicable and can be removed.

10. Conclusion and Future Work

We have implemented some of the methods proposed in this paper. More information and the source code of the implementations can be found on the website:

`http://infinity.few.vu.nl/local/`

In particular, we have implemented the method from Section 7. The program automatically finds the minimal partial model $\mathcal{A}$ for the language of normalizing S-terms, and transforms the local termination problem into a global termination problem. We have formally verified the model property, and that all terms that are not in the language of $\mathcal{A}$ are non-terminating. Global termination of the transformed system has been proven by TTT2 (1.0) [KSZM09] and formally verified by CeTA (1.05) [TS09]. Thereby we have automated one of the central contributions of [Wal00].

We intend to generalize the characterization of local termination to context-sensitive rewriting [Luc98], using μ-monotonic, partial Σ-algebras; and also to top termination, using weakly extended, monotone, partial Σ-algebras [AG00, EWZ08].

Methods using transformations from certain properties, like liveness properties [Kop08] or outermost termination [RZ09], to termination usually give rise to local termination problems. That is, termination is of interest only for those terms which are in the image of the transformation. For example, we noted that the transformation in [RZ09] gives rise to a language which can be described by a partial model. Then it suffices to show completeness of the transformation to local termination, and employing Theorem 6.4 we obtain a complete transformation to global termination for free.

Acknowledgements. We than Vincent van Oostrom and the anonymous referees for valuable suggestions for improving the presentation of the paper.

References

[AG00] T. Arts and J. Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236:133–178, 2000.

[Bar84] H. P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundation of Mathematics*. Elsevier, 1984.

[CDG+07] H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree Automata Techniques and Applications. Available at `http://www.grappa.univ-lille3.fr/tata`, 2007.

[Coq] The Coq Proof Assistant. Available at `http://coq.inria.fr/`.

[Cur30] H.B. Curry. Grundlagen der kombinatorischen Logik. *American Journal of Mathematics*, 52:509–536, 789–834, 1930.

[Der81] N. Dershowitz. Termination of linear rewriting systems. In *Proc. Colloquium on Automata, Languages and Programming (ICALP)*, pages 448–458. Springer, 1981.

[Der82] N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17:279–301, 1982.

[EdVW09] J. Endrullis, R. C. de Vrijer, and J. Waldmann. Local Termination. In *Proc. Conf. on Rewriting Techniques and Applications (RTA)*, volume 5595 of *LNCS*, pages 270–284. Springer, 2009.

[EGH+09] J. Endrullis, C. Grabmayer, D. Hendriks, J.W. Klop, and R.C. de Vrijer. Proving infinitary normalization. In *Proc. Conf. on Types for Proofs and Programs (TYPES), Revised Selected Papers*, volume 5497 of *LNCS*, pages 64–82. Springer, 2009.

[EWZ08] J. Endrullis, J. Waldmann, and H. Zantema. Matrix interpretations for proving termination of term rewriting. *Journal of Automated Reasoning*, 40(2-3):195–220, 2008.

[GHW04] A. Geser, D. Hofbauer, and J. Waldmann. Match-bounded string rewriting systems. *Applicable Algebra in Engineering, Communication and Computing*, 15(3):149–171, 2004.

[GSTSK06] J. Giesl, S. Swiderski, R. Thiemann, and P. Schneider-Kamp. Automated termination analysis for Haskell: From term rewriting to programming languages. In *Proc. Conf. on Rewriting Techniques and Applications (RTA)*, volume 4098 of *LNCS*, pages 297–312. Springer, 2006.

[GTSKF04] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Automated termination proofs with AProVE. In V. van Oostrom, editor, *Proc. Conf. on Rewriting Techniques and Applications (RTA)*, volume 3091 of *Lecture Notes in Computer Science*, pages 210–220. Springer, 2004.

[HM04] N. Hirokawa and A. Middeldorp. Dependency pairs revisited. In *Proc. Conf. on Rewriting Techniques and Applications (RTA)*, volume 3091 of *LNCS*, pages 249–268. Springer, 2004.

[Klo07] J.W. Klop. New fixed point combinators from old. In *Reflections on Type Theory, λ-Calculus, and the Mind. Essays Dedicated to Henk Barendregt on the Occasion of his 60th Birthday.*, pages 197–210. 2007.

[Kop08] A. Koprowski. *Termination of Rewriting and Its Certification*. PhD thesis, Eindhoven University of Technology, 2008.

[KSZM09] M. Korp, C. Sternagel, H. Zankl, and A. Middeldorp. Tyrolean termination tool 2. In *Proc. Conf. on Rewriting Techniques and Applications (RTA)*, volume 5595 of *LNCS*, pages 295–304. Springer, 2009.

[Luc98] S. Lucas. Context-Sensitive Computations in Functional and Functional Logic Programs. *Journal of Functional and Logic Programming*, 1998(1), 1998.

[Ohl02] E. Ohlebusch. *Advanced Topics in Term Rewriting*. Springer, New York, 2002.

[PSS97] S.E. Panitz and M. Schmidt-Schauß. TEA: Automatically proving termination of programs in a non-strict higher-order functional language. In *In Proc. Static Analysis Symposium (SAS)*, volume 1302 of *LNCS*, pages 345–360. Springer, 1997.

[RZ09] M. Raffelsieper and H. Zantema. A transformational approach to prove outermost termination automatically. *Electronic Notes in Theoretical Computer Science*, 237:3–21, 2009.

[Smu90] R.M. Smullyan. *To Mock a Mockingbird*. Oxford University Press, 1990.

[TeR03] TeReSe. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[Ter08] Termination Portal. `http://www.termination-portal.org/`, 2008. Termination Competition and Termination Problems Data Base (TPDB).

[TS09] R. Thiemann and C. Sternagel. Certification of termination proofs using ceta. In *Proc. Conf. on Theorem Proving in Higher Order Logics (TPHOL)*, pages 452–468. Springer, 2009.

[Wal00] J. Waldmann. The combinator S. *Information and Computation*, 159:2–21, 2000.
[Zac78] E. Zachos. *Kombinatorische Logik und S-Terme*. PhD thesis, ETH Zürich, 1978.
[Zan94] H. Zantema. Termination of term rewriting: Interpretation and type elimination. *Journal of Symbolic Computation*, 17:23–50, 1994.
[Zan95] H. Zantema. Termination of term rewriting by semantic labelling. *Fundamenta Informaticae*, 24:89–105, 1995.
[Zan05] H. Zantema. Termination of String Rewriting Proved Automatically. *Journal of Automated Reasoning*, 34(2):105–139, 2005.